

# COMNET III

---

## Planning for Network Managers

---

Release 1.3

# CACI

### Call or Fax for an Immediate Response

**Worldwide**

CACI Products  
3333 N Torrey Pines Ct  
Third Floor  
La Jolla, California  
92037 USA  
Tel (619) 457-9681  
Fax (619) 457-1184

**Wash., DC Area**

CACI Products  
1600 Wilson Blvd  
Thirteenth Floor  
Arlington, Virginia  
22209 USA  
Tel (703) 875-2900  
Fax (703) 875-2904

**UK**

CACI Products  
Watchmoor Park  
Riverside Way  
Camberley, Surrey  
GU15 3YL UK  
Tel +44 (0) 1276 671 671  
Fax +44 (0) 1276 670 677

Copyright © 1995 CACI  
Release 1.2 June 1996

All rights reserved. No part of this publication may be reproduced by any means without written permission from CACI.

**For product information contact:**

CACI Products Company  
3333 North Torrey Pines Ct  
La Jolla, California 92037  
Telephone: (619) 457-9681  
Fax: (619) 457-1184

CACI Products Division  
Watchmoor Park, Riverside Way  
Camberley, Surrey, GU15 3YL, UK  
Telephone: +44 (0) 1276 671 671  
Fax: +44 (0) 1276 670 677

The information in this publication is believed to be accurate in all respects. However, CACI cannot assume the responsibility for any consequences resulting from the use thereof. The information contained herein is subject to change. Revisions to this publication or new editions of it may be issued to incorporate such change.

COMNET III is a trademark of CACI Product Company.

<b>1. Introduction .....</b>	<b>1</b>
1.1 COMNET III and Network Planning .....	1
1.2 Background .....	1
1.3 Description .....	1
1.4 Applicability .....	2
1.5 Availability .....	2
1.6 Overall Approach .....	2
1.7 Types Of Network .....	3
1.8 Choosing the Correct Level of Detail .....	4
1.8.1 A Backbone Network .....	4
1.8.2 A Local Area Network .....	5
1.8.3 Summary: Level of Detail .....	5
1.9 Uses of COMNET III .....	5
<b>2. Quick Start .....</b>	<b>7</b>
2.1 Getting Started .....	7
2.2 How Models are Stored .....	7
2.3 Running a Pre-Built Model .....	7
2.3.1 Starting COMNET III .....	7
2.3.2 Loading and Running the Model .....	8
2.4 Building a Simple Model .....	14
2.4.1 Simple Model Layout .....	14
<b>3. Overview of Modeling Constructs .....</b>	<b>21</b>
3.1 Overview .....	21
3.1.1 Terminology .....	21
3.2 Network Topology .....	21
3.2.1 Nodes .....	21
3.2.2 Links .....	22
3.2.3 Subnets .....	22
3.2.4 Transit Nets .....	23
3.2.5 WAN Clouds .....	23
3.2.6 Arcs and Ports .....	23
3.3 Network Traffic and Workload .....	23

TABLE OF CONTENTS

3.3.1	Scheduling .....	24
3.3.2	Applications .....	24
3.3.3	Traffic Sources .....	25
3.3.4	Call Sources .....	25
3.3.5	External Sources .....	25
3.4	Network Operation .....	26
3.4.1	Routing Algorithm .....	26
3.4.2	Transport Protocols .....	27
3.5	Simulation Control .....	28
3.6	Statistics Reporting .....	28
3.7	User Distributions .....	29
3.8	Libraries .....	29
3.9	Model Files .....	29
<b>4.</b>	<b>Modeling Constructs .....</b>	<b>31</b>
4.1	Access Point .....	32
4.2	Buffer Processing .....	35
4.3	Circuit-switched Model .....	38
4.4	Clouds .....	40
4.5	Cloud Access Links .....	47
4.6	Cloud VCs .....	50
4.7	Command: Answer Message .....	53
4.8	Command: Process Data .....	56
4.9	Command: Read File .....	59
4.10	Command: Setup Session .....	63
4.11	Command: Transport Message .....	68
4.12	Command: Wait For .....	72
4.13	Command: Write File .....	76
4.14	COMNET Baseline .....	80
4.15	Destination: Least Busy List .....	81
4.16	Destination: Multicast List .....	82
4.17	Destination: Random List .....	84
4.18	Destination: Random Neighbor .....	86
4.19	Destination: Round Robin List .....	87

4.20	Destination: Weighted List .....	88
4.21	Distributions: System .....	90
4.22	Distributions: Table .....	92
4.23	Distributions: User .....	94
4.24	External Model Files .....	95
4.25	External Traffic Files .....	96
4.26	Flow Control .....	97
4.27	Flow Control: Fixed Window .....	100
4.28	Flow Control: Sliding Window .....	101
4.29	Flow Control: Jumping Window .....	102
4.30	Flow Control: Leaky Bucket .....	103
4.31	Flow Control: SNA Pacing .....	104
4.32	Flow Control: TCP/IP Window .....	105
4.33	IEEE 802 .....	106
4.34	Link .....	107
4.35	Link: Binary Exponential Backoff Parameters .....	109
4.36	Link: Framing Characteristics .....	111
4.37	Link: ISDN and SONET Parameter Sets .....	113
4.38	Link: Loading .....	114
4.39	Link: Aloha .....	121
4.40	Link: CSMA .....	126
4.41	Link: CSMA/CA Wireless LAN IEEE 802.11 .....	131
4.42	Link: CSMA/CD .....	135
4.43	Link: Demand-Assigned Multiple Access (DAMA) .....	139
4.44	Link: FDDI and Priority FDDI .....	140
4.45	Link: Point-To-Point .....	143
4.46	Link: Polling .....	148
4.47	Link: Port .....	152
4.48	Link: Priority Token Ring and Token Ring Enhancements .....	156
4.49	Link: Token Passing .....	157
4.50	Message Source: Interarrival Time .....	160
4.51	Message Source: Packetizing Delay .....	163

TABLE OF CONTENTS

4.52	Message Source: Priority .....	165
4.53	Message Source: Size Calculation .....	167
4.54	Message Source: Size Units .....	169
4.55	Message Source: Message Text .....	170
4.56	Node .....	172
4.57	Node: Switch .....	174
4.58	Node: Processing Node .....	179
4.59	Node: Computer Group .....	194
4.60	Node: Router .....	197
4.61	Node: Router - Updated Router Library .....	206
4.62	Packet Switched Networks .....	207
4.63	Parameter Sets .....	209
4.64	Penalty Tables .....	211
4.65	Protocol Rate Controls .....	216
4.66	Protocol Rate Controls: Available Rate .....	217
4.67	Protocol Rate Controls: Constant Rate .....	218
4.68	Protocol Rate Controls: Throttled Rate .....	219
4.69	Protocol Rate Controls: Variable Rate .....	220
4.70	Received Message Scheduling .....	221
4.71	Remotes .....	224
4.72	Reports .....	225
4.73	Routing Class: Call .....	227
4.74	Routing Class: Packet .....	230
4.75	Routing Protocol: Call .....	233
4.76	Routing Protocol: Packet .....	244
4.76.1	Max Idle Bandwidth .....	252
4.76.2	Min Delay .....	252
4.76.3	Min Queue .....	253
4.76.4	Min Sessions .....	253
4.76.5	Random List .....	254
4.76.6	Round Robin .....	254
4.77	Simulation: Animation .....	256
4.78	Simulation: Animation - Dynamic Color Change .....	259
4.79	Simulation: Parameters .....	260

4.80	Simulation: Tracing .....	262
4.81	Sockets .....	264
4.82	Statistics: Export File .....	266
4.83	Statistics: Link .....	267
4.84	Statistics: Message .....	268
4.85	Statistics: Monitors .....	270
4.86	Statistics: Plot Parameters .....	273
4.87	Subnetwork .....	274
4.88	Traffic Policing: ATM .....	277
4.89	Traffic Policing: Frame Relay .....	278
4.90	Traffic Source: Application .....	279
4.91	Traffic Source: Call .....	283
4.92	Traffic Source: Message .....	286
4.93	Traffic Source: Packet Flow/Packet Rate Matrix .....	288
4.94	Traffic Source: Response .....	289
4.95	Traffic Source: Session .....	291
4.96	Transit Networks .....	293
4.97	Transport Protocol .....	296
4.98	Trigger Events .....	300
<b>5. Creating COMNET III Models .....</b>		<b>303</b>
5.1	Introduction .....	303
5.2	Using the COMNET III Tool Palette .....	303
5.2.1	Object Creation Tools .....	304
5.2.2	Connection Tools .....	305
5.2.3	Selection Tool .....	305
5.2.4	Selecting a Group of Objects .....	306
5.2.5	Text Tool .....	306
5.2.6	Background Icon Tool .....	306
5.2.7	Subnetwork Node Tool .....	306
5.3	Moving or Repositioning Objects .....	307
5.4	Moving Around the Layout .....	307
5.5	COMNET III Menus .....	307
5.5.1	File Menu .....	308

TABLE OF CONTENTS

- 5.5.2 Edit Menu .....310
- 5.5.3 View Menu .....311
- 5.5.4 Layout Menu .....314
- 5.5.5 Create Menu .....318
- 5.5.6 Define Menu .....320
- 5.5.7 Simulate Menu .....322
- 5.5.8 Report Menu .....324
- 5.5.9 Library Menu .....325
- 5.5.10 Help Menu .....326
- 5.6 Program Operation Aids .....327
  - 5.6.1 Tool Tips and Status Bar Messages .....327
  - 5.6.2 Multiple Report Windows .....327
  - 5.6.3 3D View .....327
  - 5.6.4 Dockable Palettes and Toolbar .....327
  - 5.6.5 Arc Editing .....327
  - 5.6.6 Color Palette .....327
  - 5.6.7 Shapes .....327
  - 5.6.8 Bitmap Import .....327
  - 5.6.9 Export Encapsulated PostScript .....327
  - 5.6.10 Export Raster Bitmap .....328
  - 5.6.11 Background Text .....328
  - 5.6.12 Model Editing Features .....328
  - 5.6.13 Report Request Manager .....328

**6. Advanced Software Modeling Features .....329**

- 6.1 Introduction .....329
- 6.2 User Variables .....329
  - 6.2.1 Variable Scope: .....329
  - 6.2.2 Defining Variables: .....329
  - 6.2.3 Initial Values: .....330
  - 6.2.4 Assigning Values: .....330
  - 6.2.5 Querying Values: .....330
  - 6.2.6 Variable Errors: .....330
  - 6.2.7 Exclusive Access and Semaphore Behavior: .....330
  - 6.2.8 Troubleshooting: .....330
- 6.3 New Commands .....331
  - 6.3.1 Assign Variable Command: .....331
  - 6.3.2 Wait For Command: .....331
  - 6.3.3 Macro Command: .....332
- 6.4 Expressions .....333



<b>7. Automatic Parameter Iterator .....</b>	<b>337</b>
7.1 Introduction .....	337
7.2 Terminology .....	337
7.3 Specifying an Experiment .....	337
7.4 Running an Experiment .....	338
7.5 Analyzing the Results of an Experiment .....	338
<b>8. Reports .....</b>	<b>341</b>
8.1 Node Utilization .....	344
8.2 Application Delays .....	345
8.3 Received Message Count .....	346
8.4 File Warnings .....	347
8.5 Node Buffer Policy .....	348
8.6 Link Delays & Utilization .....	349
8.7 Random Access Link Performance .....	351
8.8 Link Frame Size Report .....	353
8.9 Link Utilization by Application .....	354
8.10 Link Utilization by Protocol .....	355
8.11 Message Delays For Message & Response Sources .....	356
8.12 Message Delays For Session Sources .....	357
8.13 Message Delays For Transport & Answer Commands .....	358
8.14 Message Delays For Setup Commands .....	359
8.15 Packet Statistics For Message & Response Sources .....	360
8.16 Message Delivery Report .....	361
8.17 Transport Retransmission Report .....	362
8.18 Transport Window and Packet Interval Report .....	363
8.19 Transport Timeout Report .....	364
8.20 Transport Ack Delay .....	365
8.21 Transport Assembly Interval .....	366
8.22 Transport Burst Size .....	367
8.23 Transport Packet Flag Report .....	368

TABLE OF CONTENTS

8.24	Transport Packet Size .....	369
8.25	Packet Delays For Session Sources .....	370
8.26	Packet Delays For Transport & Answer Commands .....	371
8.27	Packet Delays For Setup Commands .....	372
8.28	Blocked Call Statistics .....	373
8.29	Disconnected Call Statistics .....	374
8.30	Preempted Call Statistics .....	375
8.31	Call Statistics by Node .....	376
8.32	Call Statistics by Link .....	377
8.33	Node Utilization Statistics For Calls .....	378
8.34	Link Utilization Statistics for Calls .....	379
8.35	Setup Delays For Session Sources .....	380
8.36	Setup Delays For Setup Commands .....	381
8.37	Sessions Setup by Node .....	382
8.38	Sessions Setup by Link .....	383
8.39	Session Lengths by Setup Command .....	384
8.40	Session Lengths by Session Source .....	385
8.41	Session Blocking By Session Command .....	386
8.42	Session Blocking By Session Source .....	387
8.43	Buffer Input By Node .....	388
8.44	Buffer Input By Port .....	389
8.45	Buffer Output By Node .....	391
8.46	Buffer Output By Port .....	392
8.47	Cloud Throughput .....	394
8.48	Cloud VC Frame Delay and Burst Size .....	395
8.49	Cloud Access Link .....	396
8.50	Cloud Access Buffer Policy Report .....	397
8.51	Cloud Early/Partial Packet Discard Report .....	398
8.52	Global Traffic Command Reports .....	399
8.53	Response And Answer Destinations .....	400
8.54	Snapshot Reports and Alarms .....	401
<b>9.</b>	<b>Percentiles and Plots .....</b>	<b>403</b>

9.1	Overview of Plots and Percentiles .....	403
9.2	Statistics Request Buttons .....	403
9.2.1	Access Link Statistics .....	405
9.2.2	Virtual Circuit Statistics .....	406
9.3	Exporting Statistics Files .....	406
<b>10.</b>	<b>SIMGRAPHICS II Graphics Editor .....</b>	<b>407</b>
10.1	The SIMGRAPHICS II Graphics Editor .....	407
10.2	Starting SIMDRAW .....	407
10.3	Editor Windows .....	408
10.4	Setting Modes, Styles, Colors and Line Widths .....	409
10.5	Constructing Graphic Images .....	409
10.6	Selecting .....	410
10.7	Moving, Painting, Resizing, and Changing Priority .....	411
10.8	Editing Points .....	412
10.9	Cutting and Copying Objects .....	412
10.10	Using the Grid .....	413
10.11	Canvas Dimensions and Coordinates .....	413
10.12	The Library — Saving and Loading Objects .....	413
10.13	Managing the Library .....	413
10.14	Grouping Images .....	413
10.15	Recentering Images .....	414
10.16	Importing Text into SIMDRAW .....	414
10.17	Exiting SIMDRAW .....	414
<b>11.</b>	<b>COMNET Baseline .....</b>	<b>415</b>
11.1	INTRODUCTION .....	415
11.1.1	Overview .....	415
11.1.2	The System Architecture .....	415
11.1.3	Network Topology Information .....	416
11.1.4	Network Load Characterization .....	416
<b>12.</b>	<b>Statistical Distribution Functions .....</b>	<b>417</b>

TABLE OF CONTENTS

12.1	The Beta Distribution .....	417
12.2	The Erlang Distribution .....	418
12.3	The Exponential Distribution .....	419
12.4	The Gamma Distribution .....	420
12.5	The Geometric Distribution .....	421
12.6	The Hyperexponential Distribution .....	422
12.7	The Integer Distribution .....	423
12.8	The Lognormal Distribution .....	424
12.9	The Normal Distribution .....	425
12.10	The Pareto Distribution .....	426
12.11	The Poisson Distribution .....	427
12.12	The Triangular Distribution .....	428
12.13	The Uniform Distribution .....	429
12.14	The Weibull Distribution .....	430
12.15	User Distributions .....	431
12.16	Table Distributions .....	431
<b>13.</b>	<b>Menus .....</b>	<b>433</b>
13.1	File Menu .....	433
13.2	Edit Menu .....	434
13.3	View Menu .....	435
13.4	Layout Menu .....	436
13.5	Create Menu .....	437
13.6	Define Menu .....	438
13.7	Simulate Menu .....	439
13.8	Report Menu .....	440
13.9	Library Menu .....	441
13.10	Help Menu .....	442

# 1. Introduction

## 1.1 COMNET III and Network Planning

COMNET III, a graphical, off-the-shelf package, lets you analyze and predict the performance of networks ranging from simple LANs to complex enterprise-wide systems—quickly and easily. This new product builds on the success of COMNET II.5 and CACI's thirty-two years of experience in simulation technology.

COMNET III supports a building-block approach where the blocks are “objects” you are familiar with in the real world. You start with a library of objects that closely model the objects in your real networks, with one COMNET III object representing one or more real world objects. The COMNET III object's parameters are easily adjusted to match the real-world object.

COMNET III's object-oriented framework gives you the flexibility to try an unlimited number of “what if” scenarios. Your recommendations will be supported by an easy-to-understand animated picture of the network configuration you have selected—no programming required.

## 1.2 Background

There is increasing dependence on computing and communication networks in the day-to-day operation of all types of organizations. As these networks become larger and more complex the design and management of the system becomes an ever more challenging task. “Back-of-the-envelope” calculations are no longer a reasonable basis for the design validation of a multimillion dollar network. Elaborate spreadsheets won't do the job because of the stochastic nature of network traffic and the complexity of the total system. New technologies are always being introduced. New applications of communication and computing networks are constantly being explored. How do organization decide which combinations of technology and applications are right for them? As needs grow the needed investment and operating costs increase. As the organization places more dependence on its network to support its critical business operations, the risk of failure caused by poor network performance or availability may have serious repercussions. How can these alternatives be assessed at an early stage and the design of the network adapted to minimize risk?

System designers and network planners are increasingly looking for support tools to help them decide on the best design. They use the results from these tools to make their cases to clients and management.

CACI provides analysis tools for just this purpose. We have been supplying solutions to major blue chip companies, government organizations, universities and research institutes around the world for the last 30 years. We tackle leading edge problems and strive to provide techniques and tools that help you, our customer, to stand up to the challenges you face.

Our specialty is performance prediction through simulation. A model may be used to assess different design alternatives, or different operational policies. The analyst can explore the behavior of a proposed system without actually building it. It would be expensive, impractical, or even impossible to build 3 alternatives, pick the best, and throw the other two away. Alternatively, planned modifications to an existing system can be pre-tested through simulation without disturbing the actual network. A bank cannot take down its mainframe computer center during peak hours to test its recovery time performance.

We welcome feedback on the simulation tools we provide. Since they are often used to solve state-of-the-art problems we need to know where they succeed and where they do not. Only by having a constant dialogue with you can we keep our models current in the face of rapid technological change.

## 1.3 Description

COMNET III is a performance analysis tool for computer and communication networks. Based on a description of a network, its control algorithms and workload, COMNET III simulates the operation of the network and provides measures of network performance. No programming is required. Network descriptions are created graphically through a highly intuitive interface that speeds model formulation and experimentation.

COMNET III is integrated into a single windowed package which performs all functions of model design, model execution and presentation of results. A model is built and executed in several straightforward steps:

## 1. Introduction

- Nodes, links and traffic sources are selected from a palette and dragged into position on the screen. An option to automatically import the topology from Network Management Systems such as OpenView, NetView, and Spectrum is available.
- These elements are connected (using the connection tool) to define their interrelationships.
- The user double clicks on either nodes, links or traffic sources. A dialog box with all adjustable parameters appears and the user specifies the parameters for this particular item.
- Network operation and protocol parameters are set on additional dialog boxes accessed through the menubar.
- The model is verified and executed, after which the results are presented in various reports.

### 1.4 Applicability

COMNET III can be used to model both Wide Area Networks (WANs) and Local Area Networks (LANs). COMNET III models may contain both types of facilities in one integrated model. COMNET III can also provide detailed modeling of network node logic. A node's computers, their I/O subsystems, their databases and the applications which run on the computers can all be modeled.

By using discrete event simulation methodology, COMNET III provides realistic and accurate results. The alternative to discrete event simulation is to use traditional mathematically based analytical methods which cannot cope with the effects of random variance. The simplifying assumptions required by analytical methods ignore the effects of queuing, event interdependence and random variance when analyzing complex communication networks.

The network modeling approach used in COMNET III is designed to accommodate a wide variety of network topologies and routing algorithms. These include:

- LAN, WAN & Internetworking systems
- Circuit, message and packet switching networks
- Connection-oriented and connectionless traffic
- Static, adaptive and user-defined algorithms

A significant new feature of COMNET III is the ability to abstract portions of a network model and treat them as modular components. This capability follows from the object-oriented design of COMNET III. This new facility also allows the user to build a library of network components which can be "plugged in" and swapped at will.

### 1.5 Availability

COMNET III is available on most computer systems. These include many UNIX workstations and Personal Computers running Microsoft Windows, Microsoft NT, and OS/2. Each of these systems runs the same implementation of COMNET III. The only difference from one system to another is the appearance of the graphical user interface because COMNET III uses each system's style for windows, dialog boxes and other controls.

COMNET III models built on one machine type can be moved to another machine type and run with identical results. COMNET III is written in MODSIM II, a high-level, object-oriented simulation programming language.

### 1.6 Overall Approach

COMNET III is designed to accurately estimate the performance characteristics of computing and communication networks.

Estimating means that the network under study is described to COMNET III via data. COMNET III then executes a dynamic simulation of the network which builds a computer representation of the network and routes simulated traffic over it. Reports are produced on the measured performance of the different model elements and overall network characteristics, and are presented as the estimations of network performance.

This data, which is entered via a graphical user interface, describes:

1. The topology of the network: nodes, computer centers, connectivity, etc.
2. The workload placed on the network. This includes the applications that run on end systems and the traffic to be delivered across the network. The frequency and size of different tasks may be described statistically.
3. The protocols or rules for scheduling applications and routing traffic.

The reports produced are an estimate of the expected performance of the real network. Their accuracy is dependent on the data that has been entered to describe the network. One of the major questions is how accurate is the data and consequently how accurate are the estimates of performance.

Another factor which determines accuracy is the run-length or amount of simulation time the model is run. The length of the run determines how many random events are used to represent the statistically generated traffic. For instance, you may specify that file transfers are to be modeled, and that the file sizes are randomly picked between 10KB and 50KB. If you only run the model long enough to represent 5 file transfers then the file sizes might be 37KB, 21KB, 17KB, 11KB, 31KB which gives an average file transfer of 23.4KB versus an expected average of 30KB. However, if you run the model longer and obtain results over 1,000 file transfers, then the average file transfer size will converge on the expected average.

With run length in mind, the accuracy of the results of a simulation are normally quantified with a variation and a statistical confidence estimate. For instance, file transfers are completed with an average of delay of 10.5 seconds with an observed standard deviation of 2.3 seconds and with a 95% confidence of statistical correctness.

COMNET III can run multiple, independent replications of the simulation and generate mean, maximum, minimum and standard deviations, as well as plots and histograms of system performance. We recommend *Simulation Modeling & Analysis* (Averill M. Law, W. David Kelton. 2nd ed. New York: McGraw-Hill, 1991.) for a full discussion on the statistical treatment of simulation experiments.

Once you have built a model which produces accurate estimates of the performance of your network, you can then use the model for a variety of “what if” experiments. These are discussed further below.

## 1.7 Types Of Network

A network is taken to mean an arbitrary interconnection of computing and communication devices for voice, data, video, or other types of network traffic. These may include terminals, workstations, servers, network interface adapters, connection media (i.e. Ethernet cable, twisted pair), repeaters, bridges, gateways, routers, pads, front end processors, hubs, packet switches, PTT exchange equipment, pbx's, handsets, leased lines, satellite links, etc. Your organization will have your own particular network design which may include some or all of the equipment types listed above and possibly others that are not listed.

The goal of COMNET III is to provide the capability to include any network equipment type in the simulation. The user interface provides flexible interconnection of different devices so that you can describe your network to the system. COMNET III does not provide a list of every possible device ever built and used in a network as this would be an impossible task. Rather, it uses generic building blocks which can be parameterized to represent the devices you want to model. For instance, you may be a publisher and want to model a LAN system which has several print servers connected to it. Printing large image files is the main bottleneck in the system. COMNET III does not have a print server object, but it does have a Computer & Communications Node which can be used for the purpose of receiving and processing print jobs. Consequently, when you use COMNET III you have to look at your real network and make mappings of the devices and functions you have in your network to the COMNET III constructs, or add new components to the COMNET III model.

A computing network is a system like a Local Area Network (LAN) or a computer center. There is a population of users connected to it and they demand applications to be run either on their local workstations, remotely on a server, or on a mainframe.

A telecommunication network is generally a bearer system like a backbone network or a public service network such as an X.25 system. It may be private to the organization, or it may carry third party traffic as in a PTT or commercial telecommunications system. As with a computing network, there are users connected to the network but, from the point of view of the network operator, there is little knowledge of the end systems that are being serviced. For

## 1. Introduction

instance, an X.25 service provider does not know whether you are using a PC or a workstation or what software application you are using to send messages over the network— nor does the provider know what the recipient will do with any received information. What the service provider sees is simply a demand for traffic flow from an origin to a destination. The provider wants to fulfill this requirement as quickly and reliably as possible. Wide Area and Metropolitan Area networks fall into this category.

There is a rapid growth in internetworking. This is where computing networks are interconnected over communication networks (WAN or MAN) to provide access from one computing network to another. A major concern in the design of internetwork systems is the adequacy of data transmission rates offered by the telecommunications network. Computer to computer traffic normally expects to see a high speed LAN.

Voice networks can also be modeled with COMNET III. Trunk capacity, routing and peak loading issues can be investigated.

### 1.8 Choosing the Correct Level of Detail

Whichever type of network you are modeling, you have to pick the right level of detail in the model to answer the questions that are important to you. This is sometimes referred to as the granularity of the model. Think carefully about this aspect of modeling as it will greatly influence the degree of success that you have with simulation.

Not enough detail and you may miss some important aspect of the system's behavior. Too much detail and you will end up with a model which is larger than needed and which takes longer to run than necessary each experiment.

#### 1.8.1 A Backbone Network

Consider the backbone network for a bank. The bank's application development department is about to introduce a new application in all of the bank's branches. You know the size and frequency of transactions to be posted over the network, and you have estimates of the number of users and their geographical locations. Given this information, you would like to know whether current response times will be degraded when the new application comes on line. If this is the case, you must identify the cause of the bottleneck. In this case it could be inadequate leased line capacity or inadequate capacity in the network routers. It is important to know in advance whether potential bottlenecks can be avoided by acquiring additional leased line capacity or by upgrading routers.

A model of this situation can be constructed using the COMNET III nodes to represent the pads the branch offices are connected to, the backbone switches/routers, and the mainframe computer center. These are connected with point-to-point links to represent the leased lines in the system. The technical characteristics of the pads and routers are basically their throughput rate (switching times), while the point to point links are defined in terms of their speed. On top of this network you define a traffic load which originates at the pads to represent the user demands on the system. This traffic is defined as messages which carry the transaction data between the pads and the computer center and is specified by how often messages occur and how big they are.

Building a model like this allows you to determine if leased line capacity or router speeds will cause a bottleneck in your system. The model contains sufficient details about their performance characteristics to reasonably model them, and also sufficient information about the traffic load.

As an alternative to the model building approach you could have done a simple spreadsheet or “back-of-the-envelope” model by adding the average transaction rates from all of the branches and comparing this with the rating of the equipment in the computer center. For example, 50 transactions per second from users with 100 transactions per second capability on the computer center. The problem with this approach, however, is that it does not allow for the bursty nature of traffic. In real systems you would not have a transaction every 20 ms exactly. If there is congestion, a spreadsheet does not tell you where it will be or allow you to investigate solutions. In other words, the real world is not known for its ability to present you with events that occur at evenly spaced intervals at the known average rate. The variability of real world situations can only be handled with a true discrete event simulation model of the type provided by COMNET III.

Of course, you could also build a COMNET III model with more detail than this. You could assert that modeling down to the branch level and aggregating traffic there is not sufficient, and that you want to model every user connection to the system. This will increase the size of your model with little or no increase in accuracy of the results. You



will have much more information, but the additional information will not contribute to your knowledge about leased line or router adequacies. You would be faced with the task of collecting and entering more data into the system, you would have longer run times and you would have larger report files to examine because of all the devices you have modeled. Your model would be larger than it needs to be to provide the answers to the particular questions regarding line capacity and router speed.

If you added the requirement to examine the effect of the proposed change on individual users, then a more detailed model is justified, but it is not necessary to explicitly model every single user in the system. The approach which is often used in this situation is to model one or several users connections explicitly while aggregating the remainder of the users' connections. This gives performance estimates for the single user without overloading the model (and yourself) with detail.

### 1.8.2 A Local Area Network

At the other end of the scale from backbone Wide Area Networks, consider modeling the performance of a local area network (LAN). LANs are characterized by a high speed transmission medium (e.g., 10 MB/sec for Ethernet) interconnecting workstations, servers, printers, gateways etc. If you are running a database application over a LAN, it is often insufficient to model just to an arrival stream of inquiries with a corresponding stream of replies. While this will probably give a reasonable prediction of LAN utilization and transmission queueing delays over the Ethernet, it will not tell you what aspects of workstation or server operation are contributing to delay time for inquiries. In this case you will have to model the end systems, such as the database server, in more detail in order to see their affects on the system.

Modeling more detail on end systems in COMNET III is achieved by modeling more of the hardware characteristics of the end systems, together with the software load that is being placed on them. It is generally the case that poor server performance is caused by many users demanding concurrent processing or data retrieval on the server. If desired, COMNET III can model server performance in considerable detail. The speed of the processor (i.e., CPU cycle time) and disk access times can be specified together with the sequence of software actions that the server undertakes in response to any particular demand. For each of the software actions in a sequence, the load that it places on the system (in terms of CPU cycles) can be specified.

### 1.8.3 Summary: Level of Detail

You have to choose the appropriate level of detail for the system you are trying to model and the performance questions you are asking. COMNET III is capable of modeling on many levels, from modeling a specific subroutine running on a specific computer in a worldwide network, to aggregating the throughput capacity of the same network as a number of transactions per second.

## 1.9 Uses of COMNET III

Typical COMNET III applications include:

- **Peak Loading Studies**

Generally a network is subject to heavy levels of traffic at particular times of the day, week, month or year. If the network design can cope with this level of traffic then it can cope with the workload during other periods. The typical use of COMNET III is therefore to model these peak loading periods to gain an understanding of the stress points in the network.

- **Network sizing at the design stage**

When designing a new network some provision for growth has to be allowed for. COMNET III can be used to assess that the design meets current traffic levels, and it can be used to see what room there is in the design for system growth.

- **Resilience & contingency planning**

It is often important to know that a network design has sufficient resilience to offer a reasonable level of performance in various failure scenarios. The nodes and link components in a COMNET III model can be failed and recovered at various times in the simulation to test various contingencies that are not testable in the real system.

## 1. Introduction

- **Introduction of new users/applications**

New users and/or applications will typically add more load onto the network. It is useful to try and predict their impact before their introduction so that potential bottlenecks can be identified and resolved before a major problem appears

- **Evaluating performance improvement options**

Many networks have year on year traffic growth. This results in deteriorating network performance until the network is upgraded in some way. The various options for upgrading can be investigated in COMNET III as part of a cost vs. benefit study.

- **Evaluating grade of service contracts**

It is increasingly common practice for service level contracts to be negotiated between the network user and the network provider, even when they are part of the same organization. COMNET III can be used to analyze the performance service levels that can be attained during contract negotiation, and to predict potential problem areas as usage patterns of network components change over time.

## 2. Quick Start

### 2.1 Getting Started

The best way to become familiar with the capabilities of COMNET III is to sit down at a computer and try it out. This chapter will show you how to run a prebuilt model which comes with the COMNET III installation. Once you have run the model, you can perform experiments with the model by modifying some parts of the model and seeing the effect the changes have.

Next, we'll show how to build a very simple model. This will show how easy the basics are and how to save a model.

This chapter does not discuss installation of COMNET III on your particular machine since installation differs from machine to machine. There is an installation program for each machine type. Details are discussed in a separate installation guide which comes with the software.

Once COMNET III is installed on any system, however, its operation is identical and it will provide identical results when a model is run on any machine type.

### 2.2 How Models are Stored

A COMNET III model is stored in a file which has the extension `.c3`. At the same time, a subdirectory of the same name (without the `.c3`) is created for holding the output reports, trace files, etc. The installation program places the sample models in the same directory as COMNET III.

After a model is run, reports are written to a file called `stat1.rpt`. The report files are written in the model's subdirectory.

### 2.3 Running a Pre-Built Model

The model is called the Acme Bank model. It is a simple model of a bank ATM (Automatic Teller Machine transaction processing network. It isn't meant to be completely realistic, but it has each of the major elements of such a network.

Acme Bank is a small bank with branches in Boston and Washington and an ATM transaction processing center in New York. There are 60 ATMs in Boston and 60 in Washington. The ATMs in each area are on a 4MB Token Ring Local Area Network (LAN). Each LAN is connected to a CISCO packet switching router which is connected, in turn, by a backbone link to the router at the New York processing center. There is a backup connection between the routers in Boston and Washington through which packets can be rerouted if one of the backbone links goes down. The routers are connected over 9.6 kbps point-to-point links.

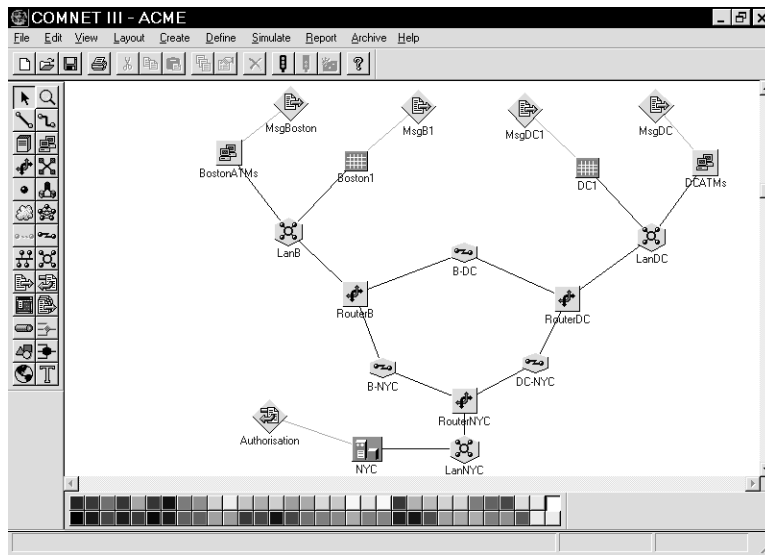
The model is setup to represent a busy peak of ATM usage where each ATM is generating a transaction once every 30 seconds. Rather than model 60 different devices in each area, we have modelled a 'composite' device which generates 120 transactions per minute (2 per ATM) by using an average interarrival time on the traffic generator of 0.5 seconds. We have also added another single node in each area (giving a total of 61 ATMs each) where we have a transaction generator with an interarrival time of 30 seconds. When we come to look at the reports we will see the response time for these single transactions as they contend with the traffic generated from the composite node.

#### 2.3.1 Starting COMNET III

On versions for Microsoft Windows double click on the COMNET III program icon.

On versions for various UNIX environments, it is first necessary to start the particular window environment for the machine type. On some machines this is the default operating mode, on others you will have to start the windowing system from the standard UNIX prompt. Open a command window and change to the directory in which you will be working. Type the command `comnet`.

## 2. Quick Start



**Acme Bank Network Layout**

The COMNET III logo is displayed while COMNET III initializes. After initialization is complete, a tool palette appears in a column along the left edge of the display and a menu bar with pull-down menus appears across the top of the display.

COMNET III will use your system's conventions for moving and resizing its window. The menu bar, scroll bars and controls conform to the standards for your system's user interface.

### 2.3.2 Loading and Running the Model

Once COMNET III is running, its File menu can be used to load the model while the Simulate menu can be used to adjust the model's simulation parameters and to start and halt the simulation.

Choose **File/Open**. Use the Open dialog box to browse to the ACME.C3 model and click on OK. The model will load and its layout appears on the screen.

To run the model, choose **Simulate/Start Simulation**.

- The tool palette on the left will be grayed out and disabled.
- The simulation clock will appear at the bottom in the status bar.
- The simulation starts and the clock starts counting time into the simulation.
- The animation reflects frames being transmitted and packets being received by nodes.

This model has been set to run with animation on and with all links operating normally. No link or node failures are scheduled during the run.

*Note to UNIX and Windows NT users: A command line interface is also available for running the simulation in non-graphic mode; by creating a batch file, one can automatically simulate a series of different models. To run a simulation in batch mode on UNIX or Windows NT, type the command*

**c3batch <model file name>**

where the **<model file name>** is the name of the **\*.c3** model file.

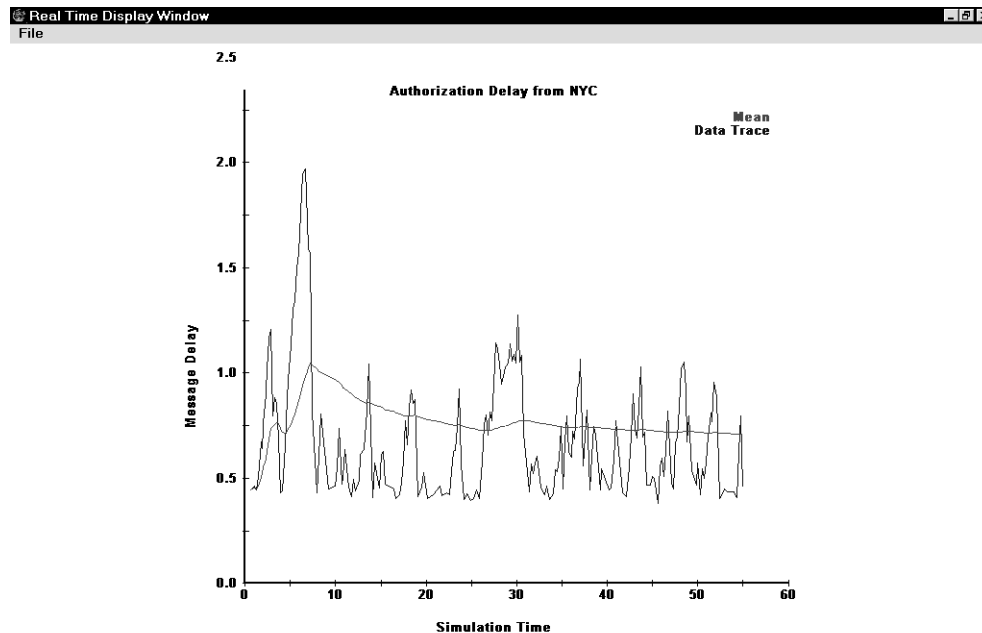
While the model is running you may

- Use the **Simulate/Animate** menu to turn animation on and off, set the speed of the animation, or schedule ani-

mation to turn on or off at a future time.

- Use the **Simulate/Trace** menu to turn tracing on or off, or to specify whether execution trace statements appear on the screen or go to a file. The trace statements appear in the status box at the bottom of the screen. You can also specify that you'd like to single step through the execution.
- Double click on any link or node icon and bring the device up or down, to simulate a link or node failure. You can also schedule a failure or recovery event at some time in the future.

As the model is running a real-time graph of authorization response times is drawn in a separate window. This graph may be expanded to full screen or closed. This real-time graph was turned on by editing the dialog box for the response source connected to the NYC node. Clicking on the **Statistics...** button at the bottom of the dialog brings up the statistics that you can turn on or off for the response source. Only the message delay is turned on here. Other statistics can be edited to turn them on or off or set their parameters.



**Real-time display of authorization delay**

Once the model has run to completion, you can examine the reports which have been written in the model subdirectory. Select **Report** from the menu bar and choose the reports you wish to view.

Note that there are no particular bottlenecks in this model. There is plenty of spare bandwidth and computer processing power. The most heavily loaded devices are the 9.6K router links at around 14% loading.

To determine the network's ability to handle a backbone link outage, bring down the backbone link labeled B-NYC or DC-NYC. This will cause traffic to be rerouted via the B-DC link.

Once the model has run again, review the reports to determine if any adjustments need to be made to the network to handle this situation.

Selected reports for the first simulation run are shown on the following pages.

## 2. Quick Start

CACI COMNET III RELEASE 1.21 Tue Mar 28 11:52:47 1996 PAGE 1

ACME

### NODE UTILIZATION

REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

NODE	DISK REQSTS GRNTED	DISK USAGE (KILOBYTES)			PROCESSOR
		AVERAGE	MAXIMUM	STD DEV	% UTIL
Boston1	0	0.000	0.000	0.000	0.05
DC1	0	0.000	0.000	0.000	0.10
RouterB	0	0.000	0.000	0.000	0.00
RouterDC	0	0.000	0.000	0.000	0.00
NYC	0	0.000	0.000	0.000	88.93
RouterNYC	0	0.000	0.000	0.000	0.00
BostonATMs	0	0.000	0.000	0.000	5.67
DCATMs	0	0.000	0.000	0.000	6.37

### INPUT BUFFER USE BY PORT

REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

NODE: CONNECTED LINKS	PACKETS		BUFFER USE (BYTES)		
	ACCEPTED	BLOCKED	AVERAGE	STD DEV	MAXIMUM
Boston1:					
LanB	1	0	0	0	101
DC1:					
LanDC	2	0	0	0	121
RouterB:					
LanB	115	0	1	9	122
B-DC	0	0	0	0	0
B-NYC	113	0	1	9	123
RouterDC:					
LanDC	130	0	1	9	122
B-DC	0	0	0	0	0
DC-NYC	129	0	1	9	122
NYC:					
LanNYC	245	0	257	234	914
RouterNYC:					
B-NYC	115	0	1	9	122
DC-NYC	130	0	1	9	122
LanNYC	242	0	2	13	123
BostonATMs:					
LanB	112	0	0	2	123
DCATMs:					
LanDC	127	0	0	4	122

## 2.3 Running a Pre-Built Model

### INPUT BUFFER USE BY NODE REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

NODE	PACKETS		BUFFER USE (BYTES)		
	ACCEPTED	BLOCKED	AVERAGE	STD DEV	MAXIMUM
Boston1	1	0	0	0	101
DC1	2	0	0	0	121
RouterB	228	0	2	13	187
RouterDC	259	0	2	13	162
NYC	245	0	257	234	914
RouterNYC	487	0	4	18	221
BostonATMs	112	0	0	2	123
DCATMs	127	0	0	4	122

### OUTPUT BUFFER USE BY PORT

#### REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

NODE: CONNECTED LINKS	PACKETS		BUFFER USE (BYTES)		
	ACCEPTED	BLOCKED	AVERAGE	STD DEV	MAXIMUM
Boston1: LanB	1	0	0	0	98
DC1: LanDC	2	0	0	0	94
RouterB: LanB	113	0	1	9	123
B-DC	0	0	0	0	0
B-NYC	115	0	15	38	258
RouterDC: LanDC	129	0	1	9	122
B-DC	0	0	0	0	0
DC-NYC	130	0	17	41	321
NYC: LanNYC	242	0	0	3	123
RouterNYC: B-NYC	113	0	14	34	123
DC-NYC	129	0	15	35	122
LanNYC	245	0	2	13	163
BostonATMs: LanB	114	0	0	2	122
DCATMs: LanDC	128	0	0	2	122

## 2. Quick Start

### OUTPUT BUFFER USE BY NODE REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

NODE	PACKETS		BUFFER USE (BYTES)		
	ACCEPTED	BLOCKED	AVERAGE	STD DEV	MAXIMUM
Boston1	1	0	0	0	98
DC1	2	0	0	0	94
RouterB	228	0	16	40	268
RouterDC	259	0	18	42	321
NYC	242	0	0	3	123
RouterNYC	487	0	31	46	235
BostonATMs	114	0	0	2	122
DCATMs	128	0	0	2	122

### RECEIVED MESSAGE COUNTS REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

RECEIVER	COUNT	MESSAGE NAME
Boston1	1	MsgB1
DC1	2	MsgDC1
NYC	112	MsgBoston
NYC	2	MsgDC1
NYC	128	MsgDC
NYC	1	MsgB1
BostonATMs	112	MsgBoston
DCATMs	126	MsgDC

### LINK DELAYS AND UTILIZATION REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

LINK	FRAMES		TRANSMISSION DELAY (MS)			% UTIL
	DELIVERED	RESENT	AVERAGE	STD DEV	MAXIMUM	
LanB	228	0	0.221	0.041	0.288	0.08
LanDC	259	0	0.218	0.040	0.286	0.09
B-DC						
FROM RouterB	0	0	0.000	0.000	0.000	0.00
FROM RouterDC	0	0	0.000	0.000	0.000	0.00
B-NYC						
FROM RouterB	115	0	74.667	16.825	101.667	14.31
FROM RouterNYC	113	0	74.668	17.128	102.500	14.06
DC-NYC						
FROM RouterDC	130	0	74.115	17.083	101.667	16.06
FROM RouterNYC	129	0	72.351	16.554	101.667	15.56
LanNYC	487	0	0.219	0.041	0.288	0.18



## 2.3 Running a Pre-Built Model

### MESSAGE DELAYS FOR MESSAGE AND RESPONSE SOURCES REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

ORIGIN / MSG SRC NAME: DESTINATION LIST	MESSAGES ASSEMBLED	MESSAGE DELAY (MILLISECONDS)		
		AVERAGE	STD DEV	MAXIMUM
Boston1 / src MsgBl: NYC	1	132.146	0.000	132.146
DC1 / src MsgDC1: NYC	2	595.881	232.242	828.122
NYC / src Authorization: ECHO	241	1168.119	562.687	2582.026
BostonATMs / src MsgBoston: NYC	112	824.877	545.192	2123.990
DCATMs / src MsgDC: NYC	128	880.905	575.863	2295.887

### PACKET STATISTICS FOR MESSAGE AND RESPONSE SOURCES

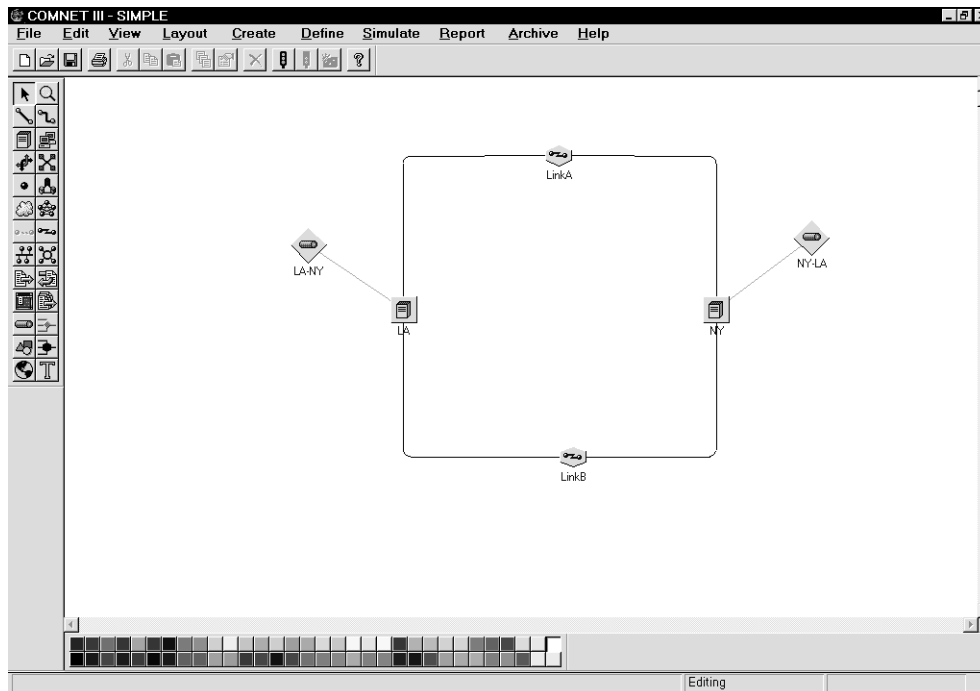
#### REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

ORIGIN: DESTINATION LIST	NUMBER OF PACKETS				PACKET DELAY (MS)	
	CREATED	DELIVERED	RESENT	DROPPED	AVERAGE	MAXIMUM
Boston1 / src MsgBl: NYC	1	1	0	0	122.146	122.146
DC1 / src MsgDC1: NYC	2	2	0	0	585.881	818.122
NYC / src Authorization: ECHO	242	241	0	0	114.303	155.949
BostonATMs / src MsgBoston: NYC	114	112	0	0	814.877	2113.990
DCATMs / src MsgDC: NYC	128	128	0	0	870.905	2285.887

## 2. Quick Start

### 2.4 Building a Simple Model

We will build a very simple model of a phone network. It will have two nodes, two call traffic generators, and two links to carry the traffic. The diagram below shows how the model should appear once we've completed the layout:



Layout of the simple model

#### 2.4.1 Simple Model Layout

Start COMNET III and choose File/New.

The palette on the left hand side of the screen allows you to create the various objects that are needed. A diagram of the palette is shown in section 5.2.

Start by clicking on the computer & communications (C&C) node tool of the tool palette. Click on the layout screen to create a node icon. Repeat the action to place a 2nd node on the screen. Leave room for the links between them. To adjust their positions, choose the selection tool at the top left of the tool palette then click on any object and drag it to a new position while holding the mouse button down.

Click on the point-to-point link tool of the tool palette and then click between the two nodes to create and position a new link. Repeat this for the second link.

Click on the source tool of the tool palette and place one call source adjacent to each node.

We need to connect the objects together using connection arcs. Double click on either of the connection tools just below the selection tool on the tool palette. This places the tool in extended mode so we can continue to use it without having to return to the tool palette to reactivate it after each use. COMNET stays in this mode until we click on the selection tool at the top left.

Click once on an object, then once on its neighbor, to place a connection arc between them.

Go back to the selection mode and double click on each icon in turn. As you double click on the icon a dialog box appears which allows you to specify details on the represented object. Give each object the appropriate name.

On the call objects, change the default interarrival time for calls from an exponential distribution with a mean of 10

seconds to an exponential distribution with a mean of 30 seconds. This can be accomplished by clicking on the button with 2 dots next to the interarrival time and then typing 30 into the mean value box.

Set the call routing algorithm for the backbone network to user defined table routing (there are no subnetworks in this model). This can be accomplished by picking the **Define/Backbone Routing** menu option and then clicking on the down arrow of the call routing protocol list box to see the choice of routing algorithms. Pick user defined table routing. We will shortly set two alternate routes between the two nodes. We need to tell COMNET III how to choose between them. Click on the parameters button next to the call routing protocol and set the primary route selection to random list. This will cause traffic to be randomly routed over the different routes, thus resulting in a balanced load sharing mode of operation. Click on OK on each dialog box to get back to the layout screen.

We now need to define the routes in the tables. Pick the LA node by double clicking on it and bringing up its dialog box. Then click on the call routing tables button. You are then presented with the following dialog box.

Call Routing from LA

Number of Routes by Destination and Routing Class

Destination	Standard		
NY	2		

You have selected routes for NY / Standard

Edit Selected OK Cancel

**Call Routing Table Selection Dialog Box**

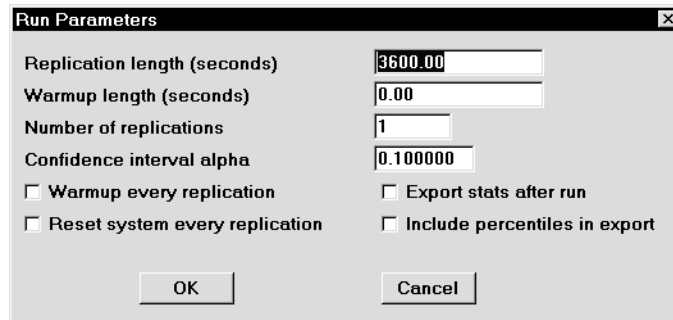
This shows that we have a possible destination of NY and the number of routes we have defined to get there (initially this is 0). Click on the number of routes box to highlight it, then click on the Edit Selected button. The 'Edit Routes' dialog box appears which is used to define the routes will appear.

The top left list box shows the list of routes that have been defined. If a route in the top left list box is picked, the links which comprise the route is shown in the top right list box. When defining a route, the possible next hops along the route are shown in the bottom right list box. On first entry to this dialog box, one empty route to get to NY is shown in the top left. The bottom right shows the list of links that could be picked. From the bottom right list pick LinkA and then pick the Add To End button underneath the top right list box. This adds the highlighted hop to the route. Since LinkA gets us completely to NY there are no more hops to add. Underneath the top left list box pick the Add To End button. This adds another route to the available route list box in the top left. At the same time the top right list box will clear (there are no hops in the new route yet) and the bottom right list box will show the possible hops that can be selected. From the lower right list box pick LinkB and then add it to the end of the route as before. This then completes the second route. Click OK on this dialog box to return to the outer dialog box, and the OK again to return to the layout screen. Go through the same process to setup routes from NY to LA.

We now need to specify how long to run the simulation for. Pick the **Simulate/Run Parameters** screen and change the run length to 1 hour (3600 seconds). Then click OK.

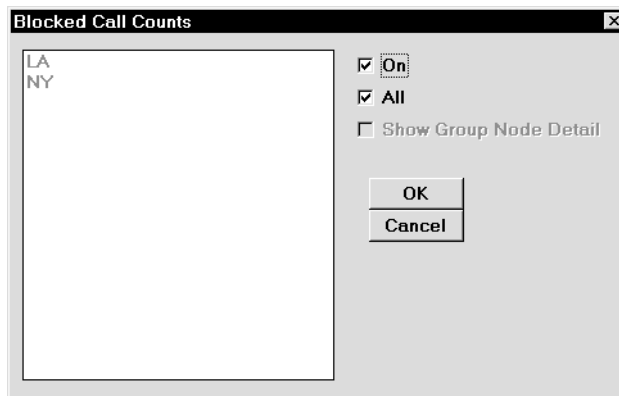
## 2. Quick Start

### Route Definition Dialog Box



### Run Parameters Dialog Box

Now pick Reports/Call Sources and select each report category: Blocked Call Counts, Disconnected Call Counts, and Preempted Call Counts and check the boxes to turn all of these reports on.



### Reports/Call Sources/Blocked Call Counts Dialog Box

To save this model, choose File / Save As, then type in the name of the model in the model name box. Do not add the ".c3" extension; COMNET III will add it.

Now select Simulate / Trace and turn trace on. Then select Simulate/Start Simulation to run the model. Note that the link's utilization appears over the link and is updated as the model is run. Also note the trace statements which appear in the bottom status box.

As the simulation is running you can pick Simulate/Animate or Simulate/Trace to turn the animation or tracing on and off. The model runs much more quickly with both animation and tracing off.

When the simulation completes you can use the Simulate/Browse Reports menu option to review the results. Note that your results may vary (slightly) depending on the order in which you built the model.

## 2.4 Building a Simple Model

### CALL STATISTICS BY NODE

REPLICATION 1 FROM 0.0 TO 3600.0 SECONDS

NODE NAME	CALLS ATTEMPTD	CALLS BLOCK AVAIL	CALLS BLOCK TRAFF	BLOCK PROB	CALLS CARRIED	CALLS DISCON- NECTED	CALLS PRE- EMPTED
LA	124	0	0	0.00	124	0	0
NY	124	0	0	0.00	124	0	0

2. Quick Start

CACI COMNET III RELEASE 1.2 Thu Feb 23 12:45:24 1996 PAGE 2

SIMPLE

NODE UTILIZATION STATISTICS FOR CALLS

REPLICATION 1 FROM 0.0 TO 3600.0 SECONDS

NODE NAME	% AVAIL	NODE FAILS	BANDWIDTH USED AVERAGE	STD DEV	(KBPS) MAXIMUM	NODE UTIL %
LA	100.00	0	670	173	1088	6.70
NY	100.00	0	670	173	1088	6.70

CACI COMNET III RELEASE 1.2 Thu Feb 23 12:45:25 1996 PAGE 3

SIMPLE

CALL STATISTICS BY LINK

REPLICATION 1 FROM 0.0 TO 3600.0 SECONDS

LINK NAME	CALLS ATTEMPTD	CALLS BLOCK AVAIL	CALLS BLOCK TRAFF	BLOCK PROB	CALLS CARRIED	CALLS DISCON- NECTED	CALLS PRE- EMPTED
LinkA	57	0	0	0.00	57	0	0
LinkB	67	0	0	0.00	67	0	0

CACI COMNET III RELEASE 1.2 Thu Feb 23 12:45:25 1996 PAGE 4

SIMPLE

LINK UTILIZATION STATISTICS FOR CALLS

REPLICATION 1 FROM 0.0 TO 3600.0 SECONDS

LINK NAME	% AVAIL	LINK FAILS	BANDWIDTH USED AVERAGE	STD DEV	(KBPS) MAXIMUM	LINK UTIL %
LinkA	100.00	0	296	106	576	19.29
LinkB	100.00	0	374	140	704	24.33

## SIMPLE

## BLOCKED CALL STATISTICS

REPLICATION 1 FROM 0.0 TO 3600.0 SECONDS

ORIGIN / CALL NAME: DESTINATION LIST	CALLS ATTEMPTD	CALLS RETRY	BLOCK PROB	HOPS AVG	HOPS MAX
LA / call LA-NY:					
NY	57	0	0.000	1.0	1
SUBTOTAL	57	0	0.000	1.0	1
LA (TOTAL)	57	0	0.000	1.0	1
NY / call NY-LA:					
LA	67	0	0.000	1.0	1
SUBTOTAL	67	0	0.000	1.0	1
NY (TOTAL)	67	0	0.000	1.0	1
** T O T A L S **	124	0	0.000	1.0	1

## SIMPLE

## DISCONNECTED CALL STATISTICS

REPLICATION 1 FROM 0.0 TO 3600.0 SECONDS

ORIGIN / CALL NAME: DESTINATION LIST	PRI	CALLS ATTEMPTD	CALLS CARRIED	CALLS DISCON	CALLS REROUT
LA / call LA-NY:	1				
NY		57	57	0	0
SUBTOTAL		57	57	0	0
LA (TOTAL)		57	57	0	0
NY / call NY-LA:	1				
LA		67	67	0	0
SUBTOTAL		67	67	0	0
NY (TOTAL)		67	67	0	0
** T O T A L S **		124	124	0	0

2. Quick Start

SIMPLE

PREEMPTED CALL STATISTICS

REPLICATION 1 FROM 0.0 TO 3600.0 SECONDS

ORIGIN / CALL NAME: DESTINATION LIST	PRI	CALLS ATTEMPTED	CALLS CARRIED	CALLS PREEMPTED
LA / call LA-NY: NY	1	57	57	0
SUBTOTAL		57	57	0
LA (TOTAL)		57	57	0
NY / call NY-LA: LA	1	67	67	0
SUBTOTAL		67	67	0
NY (TOTAL)		67	67	0
** T O T A L S **		124	124	0



## 3. Overview of Modeling Constructs

### 3.1 Overview

This overview section describes the relationships of the various COMNET III components to a simulation model. This section is divided into eight parts of a communication network simulation: Network Topology, Network Traffic And Workload, Network Operation, Simulation Control, Statistics Reporting, User Distributions, Libraries, and Model Files.

These subsections correspond to the different steps of building a COMNET III model. Typically, a Network Topology is built first, followed by adding the sources for Traffic and Workload, and setting the parameters for Network Operation. The Simulation Control parameters set up the experiment and the running of the simulation. Prior to starting a simulation, various statistics reports may be turned on for viewing during or after the simulation. User Distributions are available to bring in tabular distributions or named distributions representing a specific parameterization of a built-in analytic distribution. As the model is built, COMNET III maintains several libraries for use and re-use within the model, and there is an archive library available to allow the user to re-use objects across models. The model file contains all the information required for running the model and thus the model file is portable to other machines running COMNET III.

#### 3.1.1 Terminology

For the parameters in the COMNET III components, the following terminology is used:

Byte = 8 Bits; a byte is the smallest data unit that Comnet uses to measure message sizes.

Kilobyte (kB) = 1024 Bytes; this is the common binary definition of kilo as opposed to the metric kilo.

Kilobit (kb) = 1000 bits, and

Kilobits per second (kbps) = 1000 bits per second; These are the common metric definitions used for communications channels

Megabyte (MB) = 1024 Kilobytes = 1,048,576 Bytes

Megabits per second = 1,000,000 bits per second.

### 3.2 Network Topology

The network topology describes the layout and resources that model the physical network. The topology is defined by three basic components: nodes to represent hardware (computers or switches), links that carry traffic between nodes, and arcs to associate nodes to links. The arcs show which nodes use links, and in particular, they model the node's port connection to the link.

In addition to the basic nodes and links, there are three objects that have internal topologies: the Subnet and Transit Net for modeling of independent routing domains and hierarchical topology and the WAN Cloud for modeling wide-area network services. In either case, the icons have a detail similar to nodes or links for setting overall parameters and an internal topology for more detailed modeling. Access points (or pads) are used to mark interfaces for arcs to connect between the external and internal topologies.

#### 3.2.1 Nodes

COMNET III has four basic node types: Computer and Communications Node (C&C Node), Computer Group Node, Router Node, and Switch Node.

C&C Nodes and Computer Groups model computers. These nodes can generate or receive traffic, and they can model more complex applications concerning processor utilization and internal storage.

C&C Nodes may also model bridges, gateways, and communications switches because it can route traffic through

### 3. Overview of Modeling Constructs

itself. In contrast, Computer Groups are restricted to modeling end systems because they can only be sources or sinks for traffic.

The Router Node models hardware used for routing traffic, including routers, hubs, and switches. The Router Node is similar to the C&C Node in that it can source or sink traffic and run applications that utilize the internal processor and internal storage. However, it adds a model for an internal bus structure for moving traffic between input and output ports.

The above three nodes can be sources of traffic and resources for workload. This means that they can accept sources of traffic and applications which are described in the Network Traffic and Workload section. To model applications, these nodes include a command repertoire as well as a parameter set. Application modeling is described in the Applications Section.

In addition to the C&C Node and Router Node for routing traffic, there is an Switch node which is just for routing traffic. The Switch node models a switching fabric that is relevant for many switches that take very short times to move a packet from an input buffer to an output buffer. This node can handle any packet size. However, unlike the above nodes, this node can not be a source or sink of traffic and thus no sources can attach to this node.

The C&C, Router, and Switch nodes are also capable of modeling nodes in a circuit switched call network.

#### 3.2.2 Links

There are two classes of link models available in COMNET III: point-to-point links to represent a channel between only two nodes, and multiaccess links for local-area networks and other situations where more than two nodes share the same communications media.

Point-to-point links model communication channels between two nodes. Typically, these connections are dedicated lines such as a serial line or a dedicated line connection, particularly between routers in a wide-area or multihop networks. They are also the only type of link that can carry circuit-switched call traffic.

The WAN Cloud model provides access links to the WAN service by using a variation of a point-to-point link available inside that cloud (see WAN Clouds, sect. 3.2.4).

The multiaccess links available in COMNET III model various protocols for sharing a single medium with multiple nodes. Multiaccess protocols include Carrier-Sense Multiple-Access with Carrier Detection (CSMA/CD or ethernet), CSMA, Aloha, Token Passing (token ring, FDDI, token bus), Polling, FDDI, Priority Token Ring, CSMA/CA, Demand Assigned Multiple Access (DAMA) and ISDN and SONET parameter sets.

#### 3.2.3 Subnets

The subnet object in COMNET III is used for modeling a topology hierarchically so that separate subnets have independent and separate routing algorithms that are also independent from the backbone. The subnet icon is similar to a node in that it can only connect to links. Internal to the Subnet, the topology is similar to the top level (backbone level) and it is possible to nest subnets within subnets.

Connections between the internal topology of the subnet and the backbone topology is through the Access Points. There may be as many access points on a subnet as required. Internal to the subnet, a node attaches to an access point and that node becomes a gateway node that can route packets both at the backbone and subnet levels (or between them). Outside the subnet, the access point may be edited to access that gateway node but from the perspective of the backbone.

The subnet is used primarily for modeling interconnected subnets of independent routing algorithms. It may also be used for building a complex network hierarchically to hide detail from an upper view. Furthermore, the independent routing algorithm of the subnet often conflicts with the requirement to have a complex network that is governed by a single routing algorithm. In cases where large networks need to be modeled in a single routing domain, the topology should all be in the backbone. The view menu item "Work Area Size" is available to make the canvas area larger to contain a larger model. Using the zoom-out menu item to see the entire model, and placing items in clusters that can be zoomed into with the zoom-tool from the palette bar can organize the model similar to subnets but keep the model in the backbone routing domain.

### 3.2.4 Transit Nets

Transit Nets can be thought of as an intermediate network modelling the flow of packets through the transit net. A transit net can behave both as a link and a subnet. It is really an abstraction of a link that sends packets from the output buffer of one node connected to the net to the input buffer of another node connected to the net. Internally, the transit net behaves like a subnet and introduces an additional protocol layer at the transit net boundaries. Furthermore, the independent routing algorithm of the transit net often conflicts with the requirement to have a complex network that is governed by a single routing algorithm. In cases where large networks need to be modeled in a single routing domain, the topology should all be in the backbone. The view menu item “Work Area Size” is available to make the canvas area larger to contain a larger model. Using the zoom-out menu item to see the entire model, and placing items in clusters that can be zoomed into with the zoom-tool from the palette bar can organize the model similar to subnets but keep the model in the backbone routing domain.

### 3.2.5 WAN Clouds

The WAN Cloud is available for modeling WAN services abstractly in terms of Access Links and Virtual Circuits. An abstract model is useful when the additional detail of a more explicit model of the service is either unnecessary or unavailable.

Topologically, the WAN Cloud is similar to a link in that its icon can only connect to nodes. The WAN Cloud has an internal topology consisting of just Access Links and Virtual Circuits. Virtual Circuits are abstract models of sets of switches and links and connect to Access Links only: the arc connection to an Access Link appears as a dashed line to distinguish this abstract path from the physical connections used in the backbone topologies. The Access Link is a special case of a Point-To-Point Link that works with the WAN Cloud.

Outside of the WAN Cloud, the connection between a node and the WAN Cloud is through the Access Point which represents an Access Link in the internal topology of the cloud. Only one node can connect to an Access Point of a WAN Cloud and only one Access Link can connect to an Access Point from inside the WAN Cloud.

Virtual Circuits may be placed between any two source and destination Access Links. For any source/destination access link pair, there can be at most one Virtual Circuit. The Virtual Circuit models are directional as represented by arrows on the dashed arcs connecting them to the Access Links. Virtual Circuits are optional inside the WAN Cloud: if a Virtual Circuit is missing between the source and destination of a frame, that frame may still be routed through the WAN Cloud.

### 3.2.6 Arcs and Ports

In the backbone and Subnet topologies, the arcs connecting the links to nodes represent the connections at the nodes that interface to the link media. These arcs are editable to set parameters for the port (delay, buffers, and line-card ID) and for the link (penalty tables for routing).

## 3.3 Network Traffic and Workload

The sources for network traffic and workload add the stimulus to the network topology to drive the simulation. Network traffic refers to the messages that are sent between nodes in the topology, and workload refers to internal activities of the node’s processors or busses.

There are several kinds of traffic sources available in COMNET III. Application sources execute commands that introduce either traffic into the network or workload inside the node. Message, Response, Session & Call sources simpler sources that generate traffic between nodes. Because nodes may have processing requirements for traffic moving between them, traffic sources and commands will also implicitly add workload to the nodes. In contrast, the workload commands can only delay traffic by utilizing the processor when the traffic needs to use it.

During a simulation, COMNET III can show animation of the traffic as moving tokens representing frames and packets entering and leaving the links. The animation is along the arcs which represent the ports or connections to links; thus the animation occurs in zero elapsed time to show merely the entry or exit of the token. The delay of a link occurs when there is no animation—when the packet is waiting in a buffer or being processed inside the node or when

### 3. Overview of Modeling Constructs

the frame is transmitting or propagating inside the link.

#### 3.3.1 Scheduling

There are three methods in which sources may be scheduled: by Iteration time or by received message text, or a Trigger. There may be as many sources attached to a node as necessary, and sources connected to a node may have different or identical scheduling methods. If multiple sources are triggered simultaneously, then the commands will enter a queue at the node to be served by the processor one at a time.

Time scheduling allows sources to be scheduled according to an interval from the previous arrival, and there are options to have a source start or stop at specific times in the simulation. In a simulation model, there has to be at least one time triggered source to get the simulation to start. The Interarrival times of the sources in the backbone or in a subnet can be scaled by a Traffic Time Scale parameter in the Backbone or Subnet detail.

Scheduling by received message text provides a method to schedule sources to be dependent on messages received at the node. The message text is a label attached to a message at the source or command that sends that message. The purpose of the message text is to trigger a desired source at the destination—the message text usually does not represent the content of the message. The source that is triggered by received message may require multiple message texts and/or multiple instances of a message text before it is triggered. A wild card symbol is available to allow triggering by classes of messages.

At the destination node, there may be several sources connected that will be waiting for the same received message text; in this case, one source will be active and get the next instance of the message until it is triggered. After that source is triggered, future instances of that message text will be used to trigger the other sources in round-robin order. Usually, however, there will be just one source, if any, waiting for a particular message text.

#### 3.3.2 Applications

Applications provide a flexible means to model both workload and traffic generation at a particular node. The application is built in three parts: the node parameters for specifying the processing and storage-device speeds, the command repertoire for specifying the various actions that can occur at the node, and the application source to schedule a sequence of commands.

Workload at a node is controlled by the physical parameters for processing and storage-device times. By default, these times are zero, thus representing infinitely fast processors and storage. To get workload into a node, the parameters have to be set to non-zero. Parameters are available for processing time per cycle used by the Process command, storage parameters for the Read and Write commands, and packet processing times for traffic going through, originating, or terminating at this node.

The traffic commands include the Transport Message Command, Setup Session Command which also transmits the messages sent within that session, Answer Message Command, and Filter Message Command.

The Transport Message Command sends a single message to a single destination or a set of destinations. The messages from a Transport Command are always sent as datagrams where each packet of the message is routed independently.

The Setup Command initializes by sending a setup packet and receiving a confirmation packet. After the session is established, a number of messages may be sent during session and these messages are specified inside the Setup Command. Messages sent within a session may be routed as datagrams or as virtual circuits depending on the routing algorithm of the backbone or subnet containing the node for executing the command.

The Answer Message Command is a special command that sends its message back to the sender of the message that triggered the application source, thus the Answer Command can only be used in sources that are triggered by received message. The Answer command message is routed either as datagram or virtual circuit, copying the method used by the triggering message.

The Filter Message Command causes the node to suspend all operations until it receives an appropriate message text string defined by the Filter Message Command parameters. Once the message text has been received by the node, the Filter Message Command terminates, and the next command in the Application Source executes.

Workload is introduced to the node implicitly by the traffic through the node's packet processing parameters. In addition, there are three workload-specific commands.

The Process Command is for modeling internal calculation that makes the node's processor busy for a certain amount of time.

The Read and Write Commands also make the processor busy, but this is in the context of reading and writing files (for example, the time required for the operation depends on the size of the file). In addition to adding workload to the node's processor, the Read and Write Commands will modify the storage space which may be monitored for file space used.

The Application Source contains a sequence of commands that are executed in sequence when the Application Source is triggered either by time or by received message text. Typically, the command sequence of the Application Source is filled in after the Command Repertoire of the node has been modified to include a desired command. When an Application Source is connected to a node via an arc, the list of commands available in the Command Repertoire of the node will be available in the choices for selection when adding a new command to the sequence. The command sequence can contain commands that are either in the Command Repertoire of the node it is attached to, or in the list of Global commands available from the Define menu. Commands may be reused in command sequences in several sources or multiple times in the same command sequence.

#### 3.3.3 Traffic Sources

The traffic sources are simplifications of the applications since the single traffic source contains all the information for specifying the source. In fact, a traffic source is a special application source whose command sequence consists of a single traffic command.

The Message Source is the combination of an Application Source with a Transport Message Command and is used for modeling specific user or protocol-control messages.

The Response Source is the combination of an Application Source with an Answer Message Command and is used for modeling replies or acknowledgments to messages.

The Session Source is the combination of an Application Source with a Setup Command and is used for modeling sessions of multiple messages, bursts of messages, or messages that are routed in virtual circuits.

#### 3.3.4 Call Sources

The Call Source is the source to use for modeling circuit-switched calls. The Call Sources specify calls in terms of inter-arrival times, durations, and (through the routing class) the bandwidth requirement. In COMNET III both circuit-switched traffic and packet-switched traffic can be modeled and both may be modeled in the same simulation. However, the circuit switch model only uses the C&C Node, Router Node, ATM Switch Node and the Point-To-Point link. The resources for circuit-switched traffic are separate and independent from the data traffic resources.

#### 3.3.5 External Sources

In addition to source objects, traffic can also be introduced into COMNET III through external traffic files by using the External Traffic menu item in the Define menu. The external traffic file is a formatted text file containing a record for each traffic event: each record contains information about the time of the event, the source and destination, and other information. The traffic file may come directly from various network analyzers or it may be created from some other tool. COMNET III can interpret the events in the file as being either messages, sessions, or calls. The additional information that COMNET III requires is specified with the Edit External Sources button of the same dialog box and finally the sources are mapped to messages in the file through the Map Message ID To Message Sources button.

The AutoBaseliner utility is used to read in external traffic files and format them into an intermediate file for COMNET III use. This utility will allow multiple traffic sources to be merged into a single intermediate event file. Furthermore, the timing of the events in the external file may be scaled to speed up or slow down the recorded traffic for the simulation.

### 3. Overview of Modeling Constructs

## 3.4 Network Operation

The network operation specifies how messages are routed through the network with a Routing Algorithm and transmitted through the network with a Transport Protocol.

### 3.4.1 Routing Algorithm

Each Subnet, Transit Net, and the Backbone have separate and independent algorithms that are specified by the respective detail dialogs. In addition, there are separate choices for packet routing and for call routing which are handled independently by COMNET III. The detail for the routing algorithm also includes a “Connection-Oriented Routing for Sessions” checkbox for using virtual circuits for data traffic, or a “Preemption” checkbox for allowing call preemption for call traffic.

Several routing algorithms are available in COMNET III. There are static algorithms that compute the routing tables just at the start of the simulation and whenever a link or node fails (or recovers). There are also dynamic algorithms that periodically update the routing tables based on dynamic measures that are monitored during specified intervals. In addition, some routing protocols require Penalty Tables to be applied to each port arc (a connection between a node and link) to assign penalties for each direction of traffic into the link. COMNET III uses a convention of "from a node to a link" to establish traffic direction for the penalty tables. For each routing class, Penalty Tables specify the different penalties that class incurs when moving through a particular arc specified in the Penalty Table. Note that the default Penalty Tables assigned to the links have a penalty of 1 for each hop, thus making any algorithm using the Penalty Tables to revert to minimum hop routing unless additional tables are used.

The routing tables are created or updated based on that route or set of routes that have the least penalty. The distinction between the routing algorithms concern how the penalties are computed. The routing tables contain a route or set of routes for each triplet of source-node, destination-node, and Routing Class. Each message has a Routing Class set by the message's source or command. The Routing Class specifies some routing properties for this class of message.

By default, the routing algorithms will select a single route for each source-destination-Routing Class triplet, even if multiple routes are available with identical penalties to the minimum penalty (in which case, COMNET III will route all the traffic on the first route found). Load balancing across multiple shortest paths is available through the “Deviation % for multiple shortest paths” value-box in the detail for the selected routing algorithm: if this value is zero then all traffic for a triplet will be routed on the first path found with the minimum penalty.

For packet switched (data) traffic there are six routing algorithms available:

- RIP Minimum Hop algorithm assigns a penalty of 1 for each hop and thus selects the route that requires the fewest number of hops. This is a static algorithm that does not use Penalty Tables at links.
- Shortest Measured Delay assigns penalties to links based on the measured delay that is the sum of the output buffer delay, the transmission delay, and packet delay for each link on the path.
- Link-State Shortest Path First uses the Penalty Tables assigned to the various links of paths for penalties. It is a static routing algorithm but does require setting up the Penalty Tables for each link.
- Minimum Penalty uses the Penalty Tables which for each Routing Class may have different penalties for different levels of congestion (measured as delay or utilization). Due to the congestion thresholds, this algorithm is dynamic and has a routing update interval.
- IGRP uses a penalty that is a sum of different penalties for available bandwidth, utilization, and delay penalty from the Penalty Table. This algorithm is dynamic.
- User Defined Routing Tables allow the user to specify arbitrary routes through the network with user-selected methods for choosing among the possible alternatives.

For circuit-switched (call) traffic there are three routing algorithms available:

- Minimum Hop algorithm assigns a penalty of 1 for each hop and thus selects the route that requires the fewest number of hops. This is a static algorithm that does not use Penalty Tables at links.
- Minimum Penalty uses the Penalty Tables which for each Routing Class may have different penalties for differ-

ent levels of link utilization. Due to the congestion thresholds, this algorithm is dynamic and has a routing update interval.

- User Defined Routing Tables allow the user to specify arbitrary routes through the network with user-selected methods for choosing among the possible alternatives.

### 3.4.2 Transport Protocols

The Transport Protocol controls how the network delivers a message from the source to a destination. At the source node, a message will have a particular transport protocol. This Transport Protocol will set the packet size for the message, overhead in the packet, the type of flow control to use, and when to retransmit blocked packets. The network then routes each packet from node to node until it arrives at its intended destination.

COMNET III's Transport Protocol models transport layer protocols in terms of a routed-protocol identifier, the end-to-end packet size, a flow control method, and a retransmission interval for errors or lost packets. COMNET III maintains a list of transport protocols. Each Transport Protocol entry represents a different combination of transport layer parameters. To model a particular suite of protocols such as TCP/IP, IPX, DECNet, SNA, etc., the Transport Protocol list may be built up to include a set of transport protocol entries to represent different situations.

The routed-protocol identifier is a name that classifies a class or suite of protocols that will incur like delays in routers. For instance, IP packets will have similar processing requirements at routers even though the other parameters of the transport protocol may vary for many conditions. The routing node, then, uses the routed-protocol identifier field to determine the speed of processing the packet. Many transport protocol entries representing a suite of protocols may have the same identifier. This field is especially important for modeling multiprotocol routers that will require different amounts of processing depending on the protocol being routed. In the physical world, different protocols will impose different amounts of processing at the routing nodes and this may be modeled in COMNET III with the routed protocol identifier used to select different delays at the routing nodes.

The end-to-end packet created by the Transport Protocol determines the maximum size of the data that can be carried in a single packet. The transmitted packet will additionally include overhead bytes that represent the header and trailer bytes – in other words, all the bytes of a packet that are not part of the desired message.

The packets will be sent according to a flow control method. COMNET III provides fixed window, sliding window, SNA-pacing window, no flow control, enhanced models of these algorithms, plus an algorithm specific to TCP/IP's congestion window. X.25 networks and LAN protocols typically use fixed window protocols while many routing protocols such as TCP/IP use sliding window protocols. The window protocols have parameters for the size of the window and the size and priority of the acknowledgments required to implement the window. The enhanced versions of the windowed protocol introduce choices of error control methods (including selective repeat or re-transmitting the full window), options for delaying the acknowledgement, and for adjusting the re-transmission time based on measured round-trip time and multiple re-transmissions. The simpler versions are available for compatibility with earlier releases of COMNET III, but they may also be useful to save simulation processing and memory requirements when the enhanced features are not needed in the model.

The windowed flow control algorithms use a window size specified in terms of a number of packets. This specification is still relevant for some protocols such as TCP/IP that use window sizes in bytes because the specific algorithm restricts the packet window size to an integer number of full sized packets.

COMNET III provides sockets to model the negotiation of maximum packet and window sizes supported at both the source and destination nodes. These sockets are icons to which sources may be attached and to which traffic can be destined. The sockets have parameters such as maximum packet and window size that are used to modify the protocol's packet and window size. The socket also provides parameters for determining how packets are packetized at the sources attached to this socket, or assembled for traffic destined to this socket.

For modeling frame-level services (such as frame-relay and ATM), the protocols have two additional algorithms: a traffic policing algorithm and a traffic shaping or rate control algorithm. The traffic policing algorithm is used to flag certain packets as discard eligible (which are preferentially discarded at a congested node) and for immediately discarding packets that exceed the traffic contract. The traffic shaping algorithms control the presentation of packets into the network so that the specified burst requirements are met. These algorithms can model specific controlling

### 3. Overview of Modeling Constructs

algorithms (such as ATM's Available Bit Rate algorithm) or it can model sources (such as phone or video) that inherently introduce packets into the network at a regular rate.

A packet may be blocked at a node due to lack of buffer space or in a WAN Cloud due to a dropped frame. If a packet is dropped, then the transport protocol can retransmit the packet to correct the error. In COMNET III, there is an option to retransmit the lost packet and a parameter to set the delay before resending the packet. All packets of a message have to be received at the destination before the message is counted as delivered or assembled.

#### 3.5 Simulation Control

After a model is ready for simulation, the commands under the Simulate menu may be used to control the simulation.

The Verify command tests the model for correctness and completeness in terms of its being ready for simulation. This command is executed automatically before the simulation is started. However, while building a model, the user may use this command alone to test the current model for correctness without starting a simulation.

The Run Parameters Dialog is where the simulation experiment is defined. The Run Parameters include the replication time for the duration of the simulation for statistics collection, the warmup period when statistics are not collected, the number of replications for the number of reports, and two checkboxes for resetting the system to empty-and-idle at the end of each replication and for running a warmup for each replication.

The Start Simulation menu item will start the simulation. This command first checks the current model to determine if it has been modified since the last save — if the model has been modified, COMNET III presents a dialog box to give the user the option to re-save the model. Prior to starting the simulation, COMNET III will verify the simulation to catch any errors in the model before the simulation starts. After the model has been saved and verified, the simulation will begin.

Before or during the simulation, the Animate... and Trace... menu items are available for setting the parameters for animation or for tracing.

Animation parameters include switches for the animation and the clock as well as a field for the token speed. During the simulation with animation on, tokens representing the frames are shown entering the link and packets leaving a link. Also, numbers may appear above nodes representing numbers of sessions established on the item, percent utilization, etc.

Trace parameters include switches to trace to the screen or to a file. When tracing to the screen, there is a choice to delay the trace message for a period of time or to single step with a button to step to the next event. The trace messages are for high-level events down to the packet level: these events include application triggering, command execution, and packet actions, but generally not at the frame level. When tracing to the screen, packet traffic will highlight the node where the event is occurring in green, and call events are highlighted to show the circuit that is either established or blocked.

In addition to the above simulation control menu items, there is also the Memory Usage menu item that is useful for monitoring the number of objects allocated by COMNET III and how much memory is used. When this item is selected, COMNET III writes a file named "memstat.txt" to the working directory. This file is useful for identifying whether the model is running out of RAM and for checking to see what aspects of the model are using the most memory.

#### 3.6 Statistics Reporting

COMNET III reports statistics in a formatted report and also from statistics buttons on various objects.

The formatted reports present statistics for a particular topic for the items selected. The Report menu lists the various reports that can be turned on, and the desired report can be turned on for specific items in the model. Selecting just the desired items and statistics helps to keep the report file small and reduces processing for statistics monitoring. The last menu item in the Reports menu is Browse Reports which invokes an editor to open the model file to show the results of a simulation. The actual report file is stored as an ASCII file in the subdirectory with the same name as the model and in files "report.n" where n is the number of the replication.

In addition to reports, there are statistics that are collected for specific objects through a "Statistics..." button. Statis-



tics buttons are available on Links, Traffic Sources, and Virtual Circuits. Depending on the type of object, there are a variety of statistics monitors available. For each monitor, there is the option to collect basic statistics (minimum, maximum, mean, and standard deviation) or collect observations for plotting after the simulation is complete. When observations are saved for post-run plotting and analysis, histograms and percentiles are also available. Real-time plot monitors are also available for plotting a variable while the simulation is running. When a statistics monitor is selected in the statistics button dialog, that monitor may be edited to turn it on or off, or viewed to see the basic statistics or plots.

### 3.7 User Distributions

COMNET III makes available several analytic distributions for parameters that are random variables. This list of available distributions may be extended to include new distributions through Distribution Tables or by having Named Distributions reference a specific set of parameters for a particular analytic distribution.

A Named Distribution is a distribution with a label and a specific set of parameters for a particular analytic distribution. Named Distributions are very useful for managing complex models because all distributions are available from the Named Distributions menu and a Named Distribution may be used in many places.

A Distribution Table is also referenced by name, but it consists of a table of values and corresponding probabilities to represent either discrete or continuous distributions. To generate random values that use a Distribution Table, COMNET III samples from the user-specified table.

### 3.8 Libraries

COMNET III maintains several model-specific libraries of objects. These include libraries of parameter sets for nodes and links, routing classes and transport protocols for messages, penalty tables for routing algorithms, and named distributions and table distributions. As new items are created, they are entered in libraries so that they are available for re-use within the model. These libraries are saved as part of the model so that future modification of the model has the libraries available and so that the model file has all the information required for running the model.

In addition to the model libraries described above, there is an archive library stored separate from the model so that its elements can be re-used across models and in new models created in the future. This library is read in automatically when COMNET III is loaded, and thus it becomes part of the COMNET III environment for all models. The Archive menu has a menu item for Objects which opens a dialog box for editing the various objects that are in this archive library. The libraries may be edited to add new objects library or edit existing objects elements in the library.

Editing an external library makes the modified elements available in all new models but the old models will still have the old library elements available because each model file includes a copy of the library used to create it. Modification of the archive library only affects future models.

#### 3.8.1 Enhanced Library Access

The user interface for copying objects from the library to a model has been improved in COMNET III Version 1.3 to provide more user control over the library objects included in an individual model. In previous releases, COMNET III automatically copied to a model all of the parameter sets in the library when the model was loaded, even if the parameter sets were not needed. The new user interface presents the user with 2 lists: a list of objects in the library and a list of objects in the model. the user can select which items from the library are to be copied to the model. The new dual list interface also results in a more intuitive mechanism for adding objects (such as commands) to a model based on template objects in the library.

### 3.9 Model Files

Before a model can be simulated the first time, the model must be saved to a file. There will be a save dialog to prompt for the name of the model. After being saved, the model name shows up in the context box below the menu bar in the COMNET III window. When a model is saved, the model file names is the name of the model plus the extension “.c3”, and also there will be a subdirectory with the same name as the model and this subdirectory is where the simulation output files will be written.

### 3. Overview of Modeling Constructs

COMNET III stores the model in a special file formatted for COMNET's internal use. By default, COMNET III stores the models in binary files to improve read and write performance. As a result, the model has to be exported as an ASCII file before it can be moved to a different computer type.

#### 3.9.1 External Model File (.C3E File)

In addition to the internal file format, there is an external file format for bringing models and model data into COMNET III. The external file is in a readable, easy to use format that can specify any or all aspects of the COMNET III model. This format is available for importing model data from other programs or databases through the Import and Merge menu items in the File Menu.

When selecting the Export...C3E option from the File menu, you will have the additional option whether to include attributes that still have their default values, and objects named "DEFAULT".

Omitting defaults is handy in reducing the size and clutter of the .C3E file. More importantly, omitting defaults reduces the chance that names over which you have no control will collide during export. In fact, if you uncheck the "include defaults" checkbox, then the "change block names" checkbox will automatically be unchecked, and your object names will go into the exported file unchanged.

One caveat: Opting to omit defaults will result in a .C3E file that is "defaults-sensitive". In other words, if you save such an external model file now, and in the next release of COMNET III we change some default value, then the next time the external model file is loaded, the model may behave differently.

Selecting the "Preserve original names via TRUE-NAME=" option will cause a TRUE\_NAME attribute to be emitted. During import, the TRUE\_NAME attribute informs the reader that, while the message source may be called "Email(ID23)" to avoid name collisions in the external model file, its true name is simply "Emai". TRUE\_NAME effectively reverses the effects of name changing.

## 4. Modeling Constructs

This section lists all of the components of COMNET III in alphabetical order, together with the parameters that may be specified. A description of the execution of each component is included.

In addition, description of external model files, external traffic files, and the statistics export file are included.

## 4. Modeling Constructs

### 4.1 Access Point

#### *Purpose:*

An access point is used for graphical layout purposes only and is not an actual element of the model. For subnets, an access point connects a subnetwork node to one or more backbone links. **It is treated as an extension of the subnetwork node it is connected to.** There is no distinction between the access point and the node, except that the access point is visible on both sides of the subnetwork.

For a transit net, an access point connects a transit network link to one or more backbone nodes. **It is treated as an extension of the transit network link it is connected to.** There is no distinction between the access point and the link, except that the access point is visible on both sides of the transit network.

For WAN Clouds, access points represent the points-of-presence for the data network.

#### *Creating:*

An access point can be created inside a subnetwork (or transit net or WAN Cloud) or dropped onto a subnet icon (or transit net or WAN Cloud icon) from the backbone level. Click on the Access Point tool on the tool palette and then click on the background where the access point is required. An access point automatically snaps to the edge of the window. A subnetwork or transit network may have as many access points as required. Each access point must have a subnetwork node to connect to on a one-to-one relationship. In a transit net each access point must be able to connect to a transit network link.

#### *Connectivity:*

Inside the subnetwork, an access point must be connected to a single node. The node may only connect to one access point. Similarly, inside the WAN Cloud or transit net, the access point must be connected to an access link.

Outside of the subnetwork or transit network the access point must be connected to one or more links in the backbone. Outside of the WAN Cloud or transit net, the access point must connect to one (an only one) node.

Access points cannot be directly connected to other access points either inside or outside the subnetwork or WAN Cloud.

#### *Editing:*

If the access point is picked and edited, the dialog box for the subnetwork node or WAN Cloud access link it is connected to is displayed. An access point may be dragged to a new location from either inside the subnetwork, or on the outside of the subnetwork. An access point may not be **Cut**—however, it may be selected and **Cleared**.

To edit the backbone routing table used by a gateway node in a subnetwork, edit the node via its access point in the backbone, assuming the backbone uses user-defined table-routing. To edit a routing table used by a gateway node for destinations within a subnet, edit the node from inside the subnet, assuming the subnet uses user-defined table routing.

#### *Execution:*

During model execution, traffic may originate in the subnetwork and have a destination outside the subnetwork. In this case the subnetwork algorithm will route the traffic to the “best” subnetwork access point (“best” in the sense of the selected routing algorithm of the subnetwork) and then hand the traffic over to the backbone routing algorithm for forwarding onwards.

Similarly, inbound traffic to the subnetwork will be received via an access point (as determined by the backbone routing algorithm) and then be routed from the access point to the local destination via the local subnetwork routing algorithm.

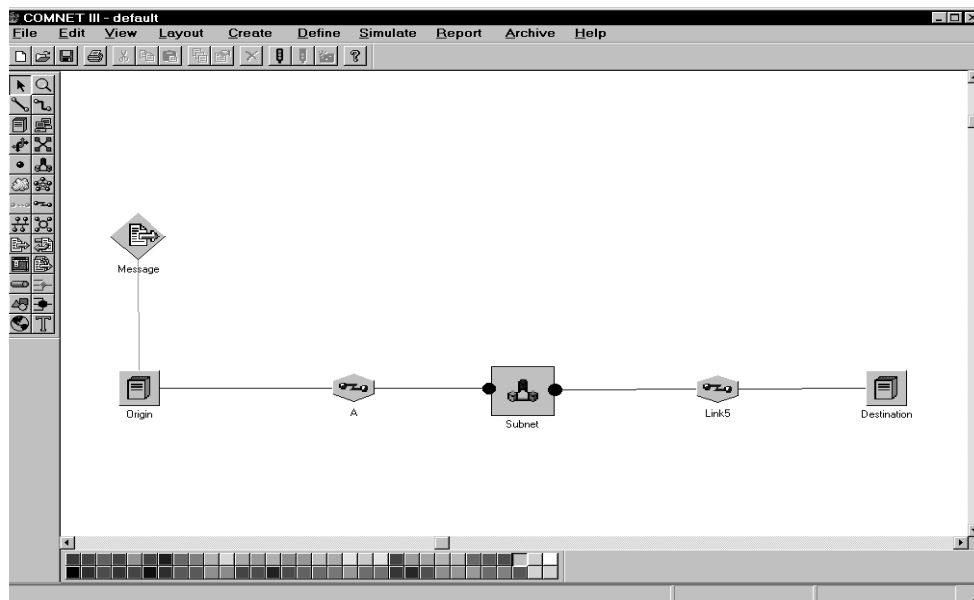
The access point will never see purely local traffic—local traffic will not route out to the backbone and back again to get to a local destination.

Traffic originating outside the subnetwork with a destination outside the subnetwork may, under certain circumstances, transit the subnetwork. This may occur in one of two scenarios.

Firstly, if the access point connects to two links in the backbone, the access point (or rather the node to which it is attached in the subnetwork) is visible to the backbone routing algorithm and so may be used as a switching point for traffic across the backbone.

Secondly, if two access points (or rather the nodes to which they are connected in the subnetwork) are directly connected in the subnetwork then the subnetwork link and access points/nodes are also visible to the backbone routing algorithm. However, if more than one hop is needed to transit between the access points in the subnetwork, then the backbone routing algorithm does not consider that a route is available via the subnetwork.

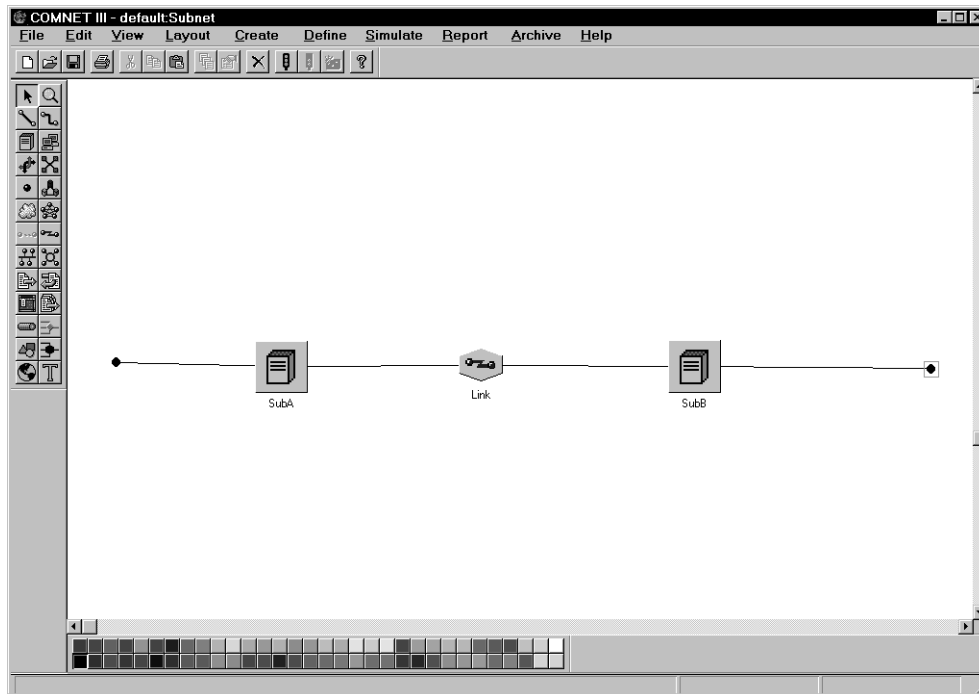
Consider the following case. A message has a destination where the only route is across a subnetwork.



**Traffic Must Transit The Subnetwork**

This traffic may be carried with the following subnetwork topology. Note that the nodes servicing the access points are connected via a single hop. If a multihop route connected these nodes then no route is visible to the backbone routing algorithm.

## 4. Modeling Constructs



### Acceptable Subnetwork Topology

**Reporting:**

The access point should be regarded as an extension of the node to which it connects in the subnetwork. The reports are therefore the node reports for the respective subnetwork node.

**Fields:**

An access point has no fields other than those associated with the node it is connected to.

## 4.2 Buffer Processing

In COMNET III, the nodes have separate buffers for input and output on each link. The buffer operation has two components -- a policy for which packets it will admit and how they will be sorted, and a processing time to model local processing that occurs before the packet in the buffer is available to leave the buffer. These buffer operations involve parameters that are on the arc connecting the node to the link for connection (or port) parameters and on the node's parameter set for common parameters.

The operation of either the input or the output buffer consists of three parts: determining whether a packet will be accepted at the buffer, ordering the packets admitted to the buffer, and processing that must occur before the packet can leave the buffer.

### *Buffer Size*

There are two size constraints for either the input or the output buffer: the maximum size available for the specific port and the maximum size available for all ports on the node. In each case, the size may be measured in terms of bytes or packets. The port parameters are available by editing the properties of the arc connecting the node to the link. The node parameters are available in the parameter set for the node.

The buffer determines that there is room for a packet by first checking that the specific input or output buffer has buffer space available to fit the incoming packet, and then it checks that the total space used for all similar (input or output) buffers on the node do not exceed the buffer constraint in the node's parameters. The packet may be accepted in a buffer if it fits in both the port-specific buffer and in the total buffer space available on the node.

This approach of having both port specific buffer size and node total buffer size is meant to be general to model different types of buffers. For example, a node may have a central buffer space that is dynamically allocated to all buffers as they need them, and in this case all of the port buffer sizes and the node's buffer size can be set to be the total size available and COMNET III will check that the total size buffered on the node will not exceed the available size on the node while allowing all the node's buffer space to be used by a single port. In another example, a node may have dedicated buffer space for each port, and in this case, the port buffer sizes can be set to be this dedicated size and the nodes buffer parameter should be set to be a large value greater than the sum of all the port buffer sizes.

Frequently, it is desirable to model the buffers with the default large value even if it is unlikely that the real node has that much buffer space. If the buffer size is unknown, COMNET III's reports on buffer utilization can give information about whether an unreasonable amount of buffer space would be required or about how much buffer space would be needed for the scenario. Another example is where the buffer size is known but the node is also known to be sized so that no blocked packets will occur and thus the user may save the step of filling out the buffer parameters because the simulation results will be the same.

### *Buffer Policies*

The above description of checking buffer size constraints is the initial test for admitting the packet into a buffer, and in particular, it describes the operation of the default buffer policy. There are two other policies available at the ports and they refine this decision: a threshold policy and a preemption policy. These poli-

## 4. Modeling Constructs

cies determine which packets may be accepted at the buffer. Generally, if a packet is not accepted in a buffer, it will be blocked and retransmitted from the source (depending on the protocol). An exception occurs for the output buffer of a switch node (which models head-of-line blocking at the input buffer): in this particular case, the packet that is not accepted waits at the input buffer until the output buffer accepts it.

In addition to determining which packets are accepted at the buffers, there are options for flagging the accepted packets for congestion to inform the end nodes that they should reduce their rate because the buffer is getting filled.

### *Threshold Policy*

Checking conformance with the threshold policy occurs after the default test so that if a packet fits in the buffer it can be rejected if the buffer size is above a threshold and the packet lacks sufficient priority to use the congested buffer. This mechanism is commonly used for rejecting packets that are marked "discard eligible" when the buffers are above their threshold in frame-switching networks such as ATM or frame-relay. The mechanism is also available for selectively admitting packets depending on whether their priority is above a minimum value allowed when the occupied buffer space exceeds the threshold.

### *Preemption Policy*

Checking conformance with the preemption policy occurs after the default test if the packet does not initially fit in the buffer. In this case, if the packet has sufficient priority, it can cause lower priority packets to be ejected from the buffer (and thus blocked) until there is enough space in the buffer for the incoming packet. The lower priority packets will be ejected only if there can be enough space made available for the incoming packet. The packets that are eligible for ejection from the buffer can either be packets of lower priority than the incoming packet or just lower in priority than the minimum priority for preemption.

This policy is also available for packets that are not marked discard-eligible to preempt packets that are already in the buffer but marked discard eligible. This may be a useful approximation for modeling more complex buffer policies on switches.

### *Explicit Congestion Notification (all policies)*

All buffer policies have options for flagging packets to inform the end nodes of congestion at this buffer. There are two forms of explicit congestion notification that may be used separately or together: forward and backward.

The forward explicit congestion notification (FECN) is set at a buffer when its level exceeds a notification threshold. This informs the destination of the traffic that there is congestion in the network. This is simple to implement but must rely on logic at the destination to then inform the source to reduce its traffic.

The backward explicit congestion notification (BECN) is set when the complementary buffer (for example, the output buffer if this buffer is the input) is above its threshold. This will inform the source of the traffic that there is congestion and thus it should reduce the traffic it is providing to the network. In COMNET III, the congestion notifications are counted in a report and they are used for some of the rate control options for the protocols.

### *Buffer Sorting*

The port buffers in COMNET III have options for how the packets are sorted once they are admitted. The packets may be sorted first-in-first-out (FIFO), by priority, or by discard-eligibility where DE packets are sorted last.



### *Processing at Buffers*

COMNET III provides an optional processing delay at the buffers to model delays for processing that may occur at a port before the packet is available to leave the buffer. This processing delay is sequential -- only one packet may be undergoing the delay at a time and so the other packet queue up waiting to be processed. If a processing delay is in effect, a packet can not leave a buffer until that packet is processed.

The processing delay parameters are available in two places: the port arc and the node. The port arc specifies the default processing delay at the port. The node parameters has a set of special case processing delays. These special case delays will override the default delay on the port arc if the special case is for the specific type of link that the arc connects and the special case is for the specific protocol ID on the protocol that created the packet.

In addition, the node's parameter set has a default port processing time. This parameter is only used to set the default processing parameters when new port arcs are added to the node. During the simulation, the default port processing delay is obtained from the port arc and the default port delay parameter on the node is ignored. If the node's default port delay parameter changes during an editing session, COMNET III will provide a dialog to allow the user to reset all the delays on the port arcs already defined.

The design goal is to have node parameters specify the processing delays at all ports so that the node can be configured and archived in the library. For pre-configured nodes, there should be no need to edit the port processing times on the port arcs unless specific information is available that indicates that the time should be different. COMNET III will pre-set the port processing times on newly added port arcs based on the node's parameter set.

## 4. Modeling Constructs

### 4.3 Circuit-switched Model

**Purpose:**

COMNET III may be used to model circuit-switched networks which carry voice, fax, video, or circuit-switched data types of traffic.

Circuit-switched traffic between an origin and destination is defined in terms of an interarrival time, a holding time, and a bandwidth requirement for the end-to-end connection.

As calls originate, a routing decision is made to reach the destination. If a route is available, bandwidth along the route is taken for the duration of the call. If no route is available (either because of link/node failure or because of congestion) the call is blocked and a reattempt optionally made some time later.

**Creating:**

A circuit-switched model is constructed by using C&C nodes as origins, destinations and switches. The connectivity is implemented using point to point links. Call sources are used to generate circuit-switched call traffic. The bandwidth requirement for a call is defined using its routing class.

**Connectivity:**

Circuit-switched traffic may only originate on C&C nodes and may only be carried over point to point links.

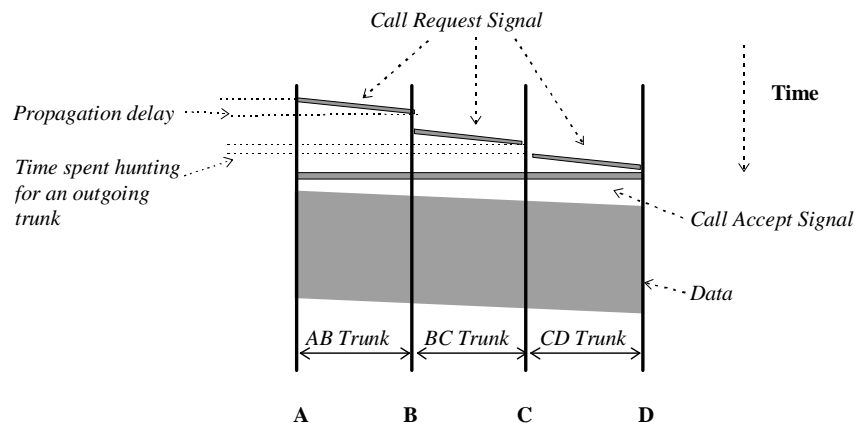
Router and ATM nodes will also switch circuit-switched traffic.

**Editing:**

See the appropriate object section (e.g., C & C, router, ATM switch, point-to-point link, call sources).

**Execution:**

When a circuit-switched call is made a call request or call routing decision must be made in order to establish the route. In a real system this will take a small amount of time, after which an end to end circuit is available for the circuit-switched traffic. The duration of the traffic (e.g. a voice call) is typically much longer than the setup delays.



In COMNET III the setup delays are ignored. When a call originates a routing

decision is instantaneously made by inspection of the routing tables currently in operation for the network. If a route is available and has the required bandwidth on all of its hops and switches, the required bandwidth will be simultaneously taken on all hops and switches and the call treated as routed.

If the call is blocked (and depending on user settings) it may be dropped, retried later, or a preemption attempt made over lower priority traffic.

***Reporting:***

Blocked Call Report  
Disconnected Call Report  
Preempted Call Report  
Calls By Node  
Calls By Link  
Utilization By Node  
Utilization By Link

***Fields:***

See the appropriate object section (e.g., C & C, router, ATM switch, point-to-point link, call sources).

## 4. Modeling Constructs

### 4.4 Clouds

*Purpose:*

The cloud provides an abstract model for public or private data networks. It provides a higher level of abstraction than is available if modeling the physical nodes and links of the network. The cloud model was designed specifically for frame relay networks, but it may also be used for modeling other networks such as X.25, TCP/IP, and cell-relay networks.

The cloud model is an alternative for modeling WANs instead of explicitly modeling the topology with routers and links. The cloud behaves like a link object in that it connects nodes and transfers frames. The cloud is different, however, in that its internal building blocks can be defined graphically with access links and virtual circuits. In routing terms, the cloud appears as a single hop such that the nodes connected to the cloud are separated from each other by one hop. The cloud model consists of the cloud icon and internal detail for access links and virtual circuits. Each node connects to the cloud through a separate access link which has its own parameters for data-rate in each direction. Double-clicking on the cloud icon or using the “enter” menu item in the edit menu will open up the cloud’s internal detail.

The cloud icon represents the common characteristics of the network, such as the framing parameters, frame dropping (or discarding) probabilities and switch delay characteristics. The virtual circuit object uses the frame dropping probabilities and switch delay to determine whether a frame gets delivered, and if it does, how long it takes to get through the network.

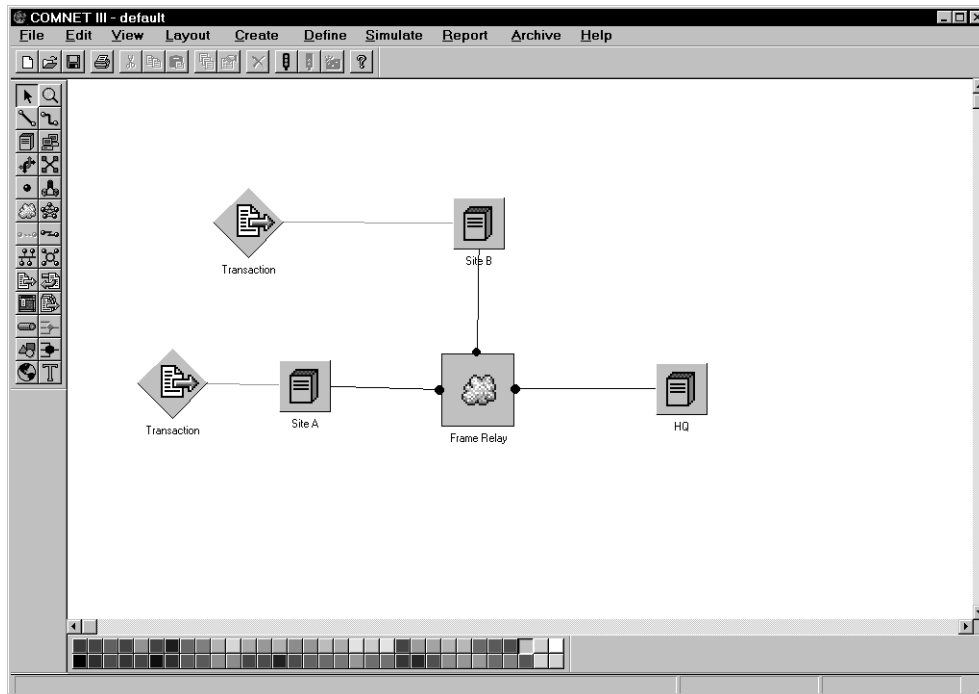
Within a real network, frames may be lost or dropped due to unrecoverable link errors, failures, lack of buffers at a router, or intentionally dropped by the service if the burst rate is exceeded. The cloud models these mechanisms as a simple probability of dropping the frame perhaps depending on the discard-eligibility (DE flag) of the frame.

For enhanced modeling, the cloud has three levels of congestion (normal, moderate, and extreme) which can change during a simulation to change the virtual circuit delay experienced by a frame and the probabilities for losing frames. The congestion changes can be scheduled randomly or periodically.

During the simulation, COMNET III will keep track of several statistics concerning the WAN performance. The statistics written to the output reports include the following: the link utilization and buffer usage for each access link, the throughput and frame-loss for both the normal or DE frames in each virtual circuit, and the frame delay and burst size for each virtual circuit.

Additionally, through the statistics button of the access links and virtual circuits, observations of many of the statistics measures may be collected for plotting during the simulation or afterwards. The observations include statistics such as average utilization, that are sampled periodically based on an average over a measurement interval.

The cloud object has many parameters to provide flexibility in modeling several aspects of WAN clouds. In order to simplify the appearance of the cloud for common models or for new users, the initial editing mode of the cloud is a basic mode with the enhanced parameters deactivated. The analyst may select an alternative editing mode that activates the enhanced parameters.



**COMNET III Network with Frame Relay Cloud**

***Creating:***

Clouds are created by choosing the cloud icon from the palette or **Create/Clouds** from the menu bar. The cloud may be placed in the model at either the backbone or subnetwork level. Double-clicking on the cloud or selecting the cloud and then using **Edit/Enter** on the menubar will open up the internal view of the cloud. The user may substitute any icon for the cloud by using **SIMDRAW** to add this icon to the `usercl.d.sg2` library file. The cloud icon is displayed behind most other network icons and thus background maps may be useful icons for clouds.

***Connectivity:***

Connections are made through access points. Outside the cloud the access points are connected to other parts of the model. Inside the cloud the access points connect to the cloud access links.

In order to connect nodes to the cloud, access points must be added to represent the access links or points-of-presence. The access point for the cloud is available on the palette bar as the black dot next to the subnet symbol. Since the access point represents a point of presence, only one node may connect to an access point. Also, a single node may only connect to one access point of a particular cloud; however, there is no limit on the number of clouds that a node may connect.

In order to prepare the cloud for simulation, access links must be defined for each access point to the cloud. The access links exist in the internal view of the cloud. In the internal view, the access link icon (the point-to-point icon) is active on the palette bar as well as in the **Create** menu under **Cloud Access Links**. Once each access point is connected to an access link, the cloud is ready for simulation by using default behaviors which would model a packet-switched, reliable-delivery network.

## 4. Modeling Constructs

### *Editing:*

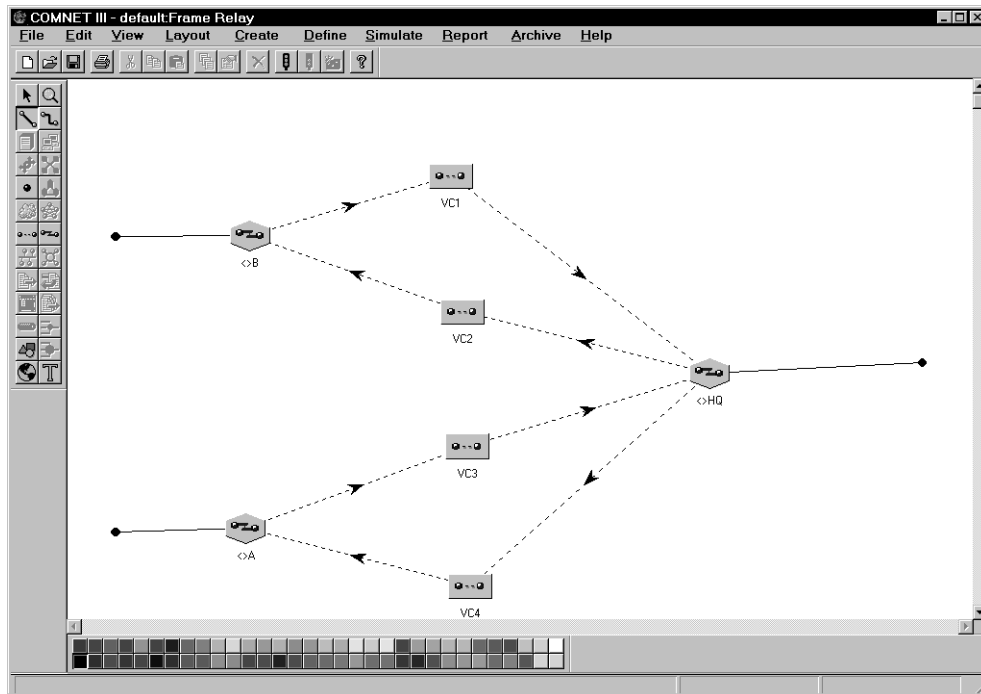
To get to the internal structure of the cloud, double-click on it or choose **Edit/Enter** from the menu bar. To edit the details of the cloud, select the cloud and then choose **Edit/Properties**. From the Cloud Detail dialog box, you can edit the parameters of the cloud and select Frame Relay or ATM, or design your own parameter set.

The first combo box provides a list of cloud types. Currently, only the WAN cloud configuration is available. This configuration provides a lot of flexibility for modeling many types of services. New configurations may be added in the future, or added through the COMNET III development environment. For each configuration there is a list of parameter sets available in the combo box below the cloud-type combo box. The parameter combo box selects the parameter set to be used for this particular cloud, and the list may be edited through the “..” button on the right side of the combo box.

Following the parameter combo box, is the parameter for specifying a statistics collection interval for internal measurements. This parameter is marked “Interval for Mean Stats (sec)” and is measured in seconds. It is used for periodic measurements of average utilization, frame delays, and sampled burst sizes within the cloud. This parameter is also available during the simulation, allowing the analyst to adjust the sampling interval as the simulation is in progress. *{This interval will “leak” expired values out of the virtual circuit leaky-bucket algorithms, and thus a periodic interval may improve the simulation performance by reducing memory requirements for expired samples. }*

The lower half of the dialog box specifies the congestion variations that will be used less frequently in models. These fields define when congestion state changes will occur during the simulation and allow the user to interactively change congestion during the simulation. There are three congestion states (Normal, Moderate, and Extreme) that are available from the bottom combo box. These congestion states correspond to different frame delays and drop probabilities set by the cloud parameters. Above the current congestion field is a parameter for specifying the next time that the congestion state will change.

The congestion state changes from Normal to either Moderate or Extreme according to the entry in the probability of extreme congestion field. Note that if only Moderate or Extreme congestion is desired in addition to Normal, then the probability of Extreme congestion will be either 0 or 1, respectively (and no random number will be drawn for either of these values). The stream number is also provided for cases where it is important to keep the probability calculation from interfering with other random number streams. If the current congestion is not Normal, then the next state change will always be to Normal.



Cloud Internal Detail Showing Access Links and Virtual Circuits

Cloud Detail Dialog Box

The parameters for the cloud type of “WAN Cloud” are available from editing the parameter list through the “..” button beside the parameter list of the Cloud Operation dialog described above. From the intermediate list box dialog, an existing parameter set may be edited or a new parameter set may be added.

## Fields

## 4. Modeling Constructs

<b>Parameter Set Name</b>	The first line of this dialog is the name of this set of parameters: this is the name that will appear in the combo box in the initial dialog box for the cloud.
<b>Session Limit</b>	The session limit refers to the total number of sessions that may be set up through the cloud at one time when connection-oriented routing is selected in the network routing parameters. This session limit is independent of the number of virtual circuits set up within the cloud; however, with connection-oriented routing turned on, the network-level routing algorithm will not route new sessions through the cloud if the cloud's session limit is exceeded.
<b>Frame min, max, overhead</b>	The frame parameters of “min”, “max”, and “overhead” specify the constraints for all frames (or encapsulation packets) created and routed through the cloud. If the frame minimum is greater than zero, then padding will be modeled so that no frame is less than the specified size. If the frame minimum, maximum, and overhead are all zero, then the frames will take on the size of the packet. If both parameters are equal, then the frames become cells (in other words, frame relay becomes cell relay because all frames are the same fixed size). The minimum and maximum sizes include the frame overhead such that the amount of packet data that fits in the frame is the frame size minus the overhead.
<b>Frame Priority</b>	The frame priority parameter is available for specifying how the frames will be treated when arriving at buffers within the cloud. The frame priority may either be none (where all frames are equal), equal to the packet priority, or related the status of the DE condition of the frame.
<b>Frame Assembly</b>	Frame Assembly allows a single frame to carry as many waiting packets or packet fragments as will fit in the frame.
<b>Use CIR in VCs</b>	The virtual circuits measure burst size with the leaky bucket algorithm. The leak-interval of the bucket may either be described directly as an interval or indirectly through a committed information rate (CIR). If the “Use CIR in VCs” is turned on, then the leak parameter is specified in the VC as a CIR, otherwise the interval is specified directly.
<b>Transmit Non-VCs</b>	VCs are optional in the cloud: the cloud will deliver frames to the destination even where a VC is not defined. The “Transmit Non-VCs” switch may be turned off to have the cloud drop frames whose destinations have no defined VCs. The default behavior with this switch on is to have default VCs which do not monitor the burst size but still delay the frame as if it had gone through an explicit VC.
<b>Routing Interval</b>	The “Routing Interval” field provides a possibly random interval between updates of the number of switches a virtual circuit contains. This behavior is only active when the “Enhanced Model” switch is turned on; otherwise, all VCs delay frames with the same delay parameter. When the routing interval is active, the VC will select a random integer for the number of switches traversed by the VC and then use that number to multiply the delay specified by the “Delay/Switch (ms)” field described below. The purpose of this parameter is to model variations in delay that occur in a real WAN as a result of varying route lengths between different access links.
<b>Delay Switch</b>	The delay that the frame will experience for each frame in a VC. This delay parameter is meant to model both the queuing delay and the switch latency at the switches within the cloud. Typically each VC will use the value drawn from this distribution to determine the delay for its current frame. If the enhanced model is



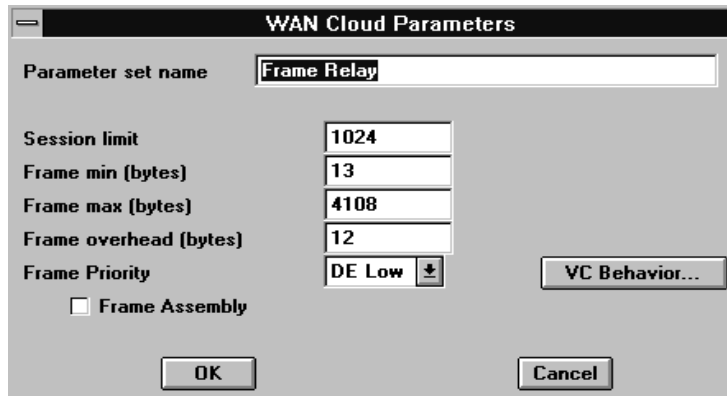
in effect, each VC will scale the delay by the number of switches that the VC specifies. The congestion level determines which distribution is used for frame delay, but these values may all be equal if the delay does not depend on congestion, and they may be constants if the variation (or jitter) in delay is insignificant to the model.

### Frame Drop Probability

The probabilities that the frames will be dropped by the virtual circuits depending on both the congestion state of the cloud and the DE state of the frame. Typically, the frames will be more likely to be dropped at higher levels of congestion or when the frames are marked DE.

*In normal conditions, the normal frame will not be dropped by the virtual circuit unless it arrives at a switch that is extremely congested and is refusing to accept any frames or one of the network's links experiences transmission errors. For a case where an extremely congested switch or a transmission error is a possibility, the probability of dropping a normal frame may be non-zero.*

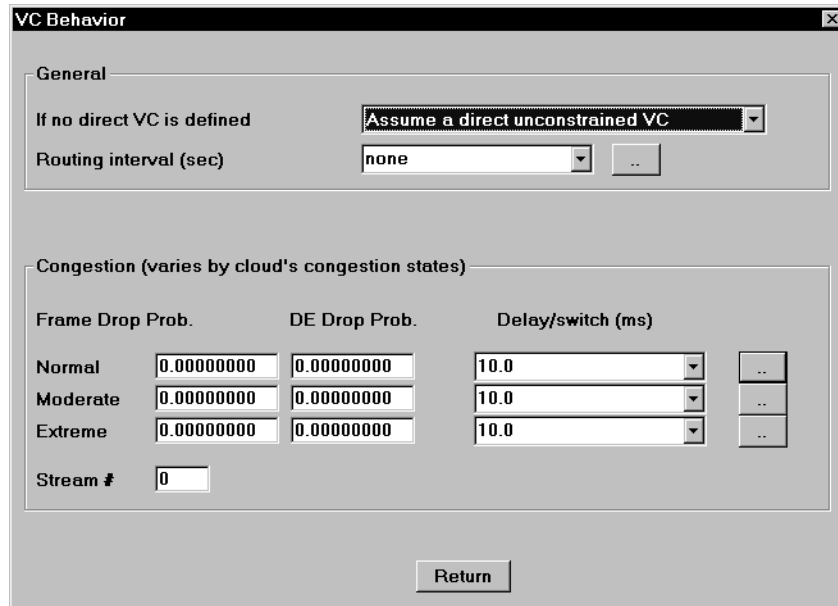
#### 4. Modeling Constructs



The dialog box is titled "WAN Cloud Parameters". It contains the following fields and controls:

- Parameter set name:
- Session limit:
- Frame min (bytes):
- Frame max (bytes):
- Frame overhead (bytes):
- Frame Priority:  with a dropdown arrow
- Frame Assembly
- VC Behavior... button
- OK button
- Cancel button

Frame Relay Cloud Parameters



The dialog box is titled "VC Behavior". It contains the following sections and controls:

- General section:
  - If no direct VC is defined:  with a dropdown arrow
  - Routing interval (sec):  with a dropdown arrow and a button with two dots
- Congestion (varies by cloud's congestion states) section:

	Frame Drop Prob.	DE Drop Prob.	Delay/switch (ms)	
Normal	<input type="text" value="0.00000000"/>	<input type="text" value="0.00000000"/>	<input type="text" value="10.0"/> with a dropdown arrow	<input type="button" value=".."/>
Moderate	<input type="text" value="0.00000000"/>	<input type="text" value="0.00000000"/>	<input type="text" value="10.0"/> with a dropdown arrow	<input type="button" value=".."/>
Extreme	<input type="text" value="0.00000000"/>	<input type="text" value="0.00000000"/>	<input type="text" value="10.0"/> with a dropdown arrow	<input type="button" value=".."/>
- Stream #:
- Return button

VC Behavior Dialog

## 4.5 Cloud Access Links

### *Purpose:*

Cloud access links model the connections between the user and the cloud: this connection may be either the link from DTE or PAD in X.25 or the link from the frame relay access device in frame relay services. Typically, this link will model the local line between the user and the network's point of presence (POP).

The access link also models the buffer at the POP switch for the exiting frames that may arrive simultaneously from many sources connected to the cloud. This buffer allows the cloud to model cases where the combined data rate from all virtual circuits momentarily exceeds the data rate for the access link. This buffer sorts the waiting frames by priority (see Frame Priority under 4.3 Clouds. Higher priority frames may preempt waiting frames with lower priority.

Important parameters of the access link include the data rate and number of circuits of the link and how much buffer space the network POP switch provides for traffic arriving at the node.

The access link collects statistics on link utilization and exit buffer size. These statistics are available for post-analysis and real-time plotting.

### *Creating:*

A cloud access link can be created while within a cloud. You can either choose the point-to-point link tool from the palette or choose **Create/Cloud Access Links** from the menu bar.

### *Connectivity:*

The pads that are available from the tool palette are the connection points from outside the cloud to the access link. Within the cloud each pad must connect to one access link and each access link must connect to one pad.

### *Editing:*

Cloud access link parameters are edited by double-clicking on the link or choosing **Edit/Detail** while the access link is selected (highlighted in red).

**Cloud Access Link Parameters**

## **Fields**

### **Number of Circuits**

The number of circuits in the access link, this is usually left at one.

### **Entry, Exit Bandwidth**

These are usually set to the same value, but can be set to different values for spe-

## 4. Modeling Constructs

cial cases. A value of zero has the special meaning of infinite bandwidth (i.e., no transmission delay).

### Propagation

Propagation time of the signal across the access link itself. This value is usually insignificant and may be left as zero.

### Exit (Egress) buffer

This a buffer that is available at the input side of the exit part of the access link. A particular access link may be connected to several virtual circuits that deliver frames at the same time, in which case the frames may have to queue to wait for the earlier frames to get transmitted. This buffer is sorted by priority of the frames (the method for determining frame priority is set in the cloud parameters).

The screenshot shows a dialog box titled "Cloud Exit (Egress) Buffer". It is divided into two main sections. The top section, "Buffer Parameters", contains a text input for "Buffer limit" with the value "0", a checked checkbox for "Use unlimited buffer", a dropdown for "Buffer units" set to "Bytes", a dropdown for "Buffer definition" set to "Per Access Link", a dropdown for "Buffer policy" set to "Default", and a dropdown for "Parameters" set to "DEFAULT" with a small "..." button to its right. The bottom section, "Early Packet Discard/Partial Packet Discard", contains a text input for "Buffer threshold (buffer units)" with the value "10000000.000" and a dropdown for "Discard type" set to "None". A "Return" button is located at the bottom center of the dialog.

### Buffer limit

The buffer limit may be specified in terms of packets or bytes. If this value is exceeded then the buffer will block incoming frames.

### Use unlimited buffer

If this box is checked there is no buffer limit and the buffer will always accept incoming frames.

### Buffer units

The buffer size can be specified in either packets or bytes.

### Buffer definition

The buffer may be separate for each VC (Per VC) or just a single buffer for the whole access link (Per Access Link).

### Buffer policy

Described earlier under *Buffers*.

### Early Packet Discard/Partial Packet Discard:

**Buffer threshold (buffer units)** Specifies at what level of buffer congestion the logic for Early Packet Discard and Partial Packet Discard will become active.

### Discard type

Early packet discard and partial packet discard are options for rejecting frames

once a previous frame from the same packet is lost. This option is particularly relevant to modeling cell-based services such as ATM that will segment a higher layer protocol packet into many small cells. Because these services do not have provisions for error corrections by retransmitting the cells, a lost cell implies that the service will not be able to reassemble the packet that that cell is carrying. In this case, the buffer that drops the cell can recognize that any further cells from the same packet need not be forwarded. This is called partial packet discard and it can avoid wasting bandwidth on unusable transmissions. This logic is further extended to early packet discard for when the first cell of a new packet arrives, the buffer can immediately begin to reject all the cells of that packet if it appears that all of the cells in the packet will not be able to fit in the buffer. The switch determines that all the cells in a packet will not fit by using a fixed threshold on the buffer based perhaps on the maximum size packet possible for the packet.

## 4. Modeling Constructs

### 4.6 Cloud VCs

#### *Purpose:*

A cloud VC represents the virtual circuit or path through the network that connects a source to a destination. In the actual network, the path may consist of a number of intermediate routers and link hops, and the virtual circuit icon models this path in terms of a delay consisting of a propagation delay, plus a delay that is the product of the specified number of switches the path contains and the delay per switch.

Multiple transits through the cloud are allowed in order to get to a destination that does not have a direct VC. COMNET III will attempt to find a way to get to the next hop destination by following the explicit VCs through multiple transits until the destination is reached. This is particularly helpful for modeling frame-relay services where remote offices communicate to each other through the cloud with a central site acting as a router.

The primary purpose of the virtual circuit object is to specify the burst constraints for traffic on this virtual circuit. Burst sizes are measured with a leaky bucket algorithm: the leaky-bucket burst is the amount of data accepted by the virtual circuit for the previous interval of time. For example with a one second interval, the burst size will be the amount of data that entered the virtual circuit over the previous second. The burst is recalculated for each frame that arrives at the virtual circuit.

The virtual circuit uses the burst size calculation to determine how to handle a frame entering the network from an access link. If the frame would result in the burst being over the committed plus excess burst size, then the virtual circuit will immediately drop the frame. If the frame would result in the burst size being over the committed burst size, then the frame will be marked DE (discard eligible); otherwise, the frame is marked normal. The DE or normal frame will be successfully transmitted to the exit access link unless it is dropped due to Frame Drop Probability (see *Clouds*).

#### *Creating:*

A cloud VC can only be created while within the cloud, either by selecting the Cloud VC icon from the palette or choosing **Create/Cloud VCs** from the menu bar.

#### *Connectivity:*

The virtual circuits connect between two access links—there can be at most one virtual circuit between a particular source and destination access link. When a connection is made, the arcs will be broken to indicate a virtual connection, as opposed to the physical connections used between nodes and links. The arrows on the arcs indicate the direction of information flow. Generally, a virtual circuit has to be defined for both directions of a pair of access links to specify burst parameters and collect statistics independently for the different directions.

The virtual circuit will measure information rate bursts using a leaky bucket, sliding window, or jumping window and mark frames according to the current burst size. This behavior is specifically designed for frame-relay service but it may be useful for modeling or measuring bursts in other services. The burst parameters include a committed burst size and an excess burst size measured over an interval that is either specified or implied by a committed information rate. The virtual circuit will (1) immediately drop frames if the burst size resulting from accepting the frame exceeds the sum of the committed and excess burst sizes, (2) mark the frame as discard eligible (DE) if the resulting burst size is greater than the com-

mitted burst size, or (3) just leave the frame unmarked (non-DE or “normal”) if the burst size is less than the committed burst size. For dropped frames, the cloud does not model error correction and thus the associated packets of the dropped frames will be blocked by cloud. For more details on burst rate measurements, see the specific sections on flow control.

**Editing:**

Cloud VC parameters are edited by double-clicking on the virtual circuit or choosing Edit/Detail while inside the cloud.

**Cloud Virtual Circuit Dialog Box**

**Fields**

**Committed Information Rate or Burst Interval**

The burst interval may be specified either as a committed information rate (CIR) or an interval depending on the setting of the switch in the cloud parameters. If CIR is specified, the burst interval is computed as the committed burst size divided by the CIR unless either value is zero, in which case the interval is assumed to be one second. The CIR is in units of kilobits per second.

**Committed burst size**

The committed burst size, measured in kilobits (1000 bits), is the threshold on the burst size, beyond which the new frames will be marked DE.

**Burst Type**

Sets the type of burst measurement used: leaky bucket, jumping window, or sliding window.

**Excess burst size**

The excess burst size, in kilobits, is the threshold on the amount in addition to the committed burst size above which all frames will be dropped immediately.

**Mark DE Frames**

This option allows you to bypass the traffic policing when it is known that a certain type of traffic will never (or always) be set discard eligible. The options for bypassing the algorithm for setting discard eligibility of packets are to always set the DE flag or to never set the flag. The "use algorithm" option uses the burst measurement algorithm with the thresholds to determine discard eligibility.

#### 4. Modeling Constructs

##### **Never Drop Due To Excess Burst**

This is a checkbox for preventing the algorithm from rejecting any excess traffic that occurs after the burst exceeds the committed plus excess burst size.

##### **Propagation delay**

The propagation delay across the virtual circuit, between entry and exit points of presence.

##### **Number of switches**

There is an additional parameter for the number of switches that this virtual circuit uses. If the enhanced model is not selected in the cloud parameters, this field will be deactivated and the number will be assumed to be one. Otherwise the virtual circuit will have a preset or random number of switches which multiplies the frame delay from the cloud parameter to get the frame delay for each transmitted frame. In the enhanced model, the number of switches may change randomly at each routing update interval in the cloud parameters.

##### **Show burst size**

This check box is for the graphical display of the current burst size of this virtual circuit. Each virtual circuit has independent control on whether the burst size will be displayed above it during simulation. The burst size will be displayed when the box is checked (even when animation is off). This check box is active during the simulation so that the display can be interactively changed during the simulation. The displayed burst size represents the sampled burst size at the measurement interval specified in the initial cloud dialog box.



## 4.7 Command: Answer Message

### **Purpose:**

An Answer Message command is used within an Application Source to send a reply back to the origin of the message which caused the application to execute. In other words, the application must have been started by received message scheduling for an answer message command to return a reply back to the received message's origin.

Answer messages are routed according to the message that triggers them: datagram if the triggering message is a datagram, connection-oriented if the triggering message is connection-oriented (session source).

If the application has several message requirements then the reply is sent to the origin of the first message in the requirement list. If the first message requirement is a wildcard, the reply is sent to whichever node sent the message which satisfied the message requirement.

### **Creating:**

Answer message commands can be created in one of three ways.

Firstly, they may be created locally on computer & communication nodes, computer group nodes, and router nodes. The respective node dialog box has a Commands button which, when pressed, brings up a dialog box for editing and creating commands. If this method of creation is used then the respective answer command only has local scope to the node on which it is defined.

Secondly, answer commands may also be created using the Define/Commands menu option. In this case, the scope of the command is global to the model. A command defined globally can be executed by any node in response to an application which runs on the node and calls the global command.

Thirdly, answer commands can be added to the library via the Archive/Object menu option and then picking the Commands line from the offered list box. In this case the new answer command is saved in a library and is then available for use across models. To use an answer command defined this way in a particular model, it must be copied out of the library file to the model by using either the first or second method outlined above. It will then be either local to a node or global, depending on the method used.

In addition, an Answer Message command is created automatically when the user creates a Response Source.

### **Connectivity:**

An answer message command must be defined on a computer & communications node, on a computer group node, on a router node, or as a global command from the Define/Commands menu. Once defined, it may be referenced in the command list of any application which runs on the node.

If a global command and a local command have the same name, then the local command will take precedence.

### **Editing:**

Edit the details of the answer command from the node dialog boxes, or from the Define/Commands menu option, or from the Archive/Objects menu option.

When the answer command is edited in the library (via Archive/Objects) the changes made are not automatically copied into the local model, or other models

## 4. Modeling Constructs

that use the library defined command. For the changes to take effect the answer command must be recopied from the library.

The screenshot shows the 'Answer Command' dialog box with the following settings:

- Name: Ack1
- Priority: 1
- Routing class: Standard
- Transport protocol: Generic
- Packetize (ms): 0.0
- Net svc level: 1
- Use "ECHO" in reports:
- Msg size calc: Probability distribution
- Prob distrib: 1000.0
- A: 1.000
- B: 0.000
- Msg size units: Bytes
- Msg text option: Use original message
- Msg text: (empty)

**Answer Command Dialog Box**

### *Execution:*

An answer command is executed by being called from an application that has been scheduled on a node. The application requests its commands to be executed one after the other. When it is the turn of an answer command to execute the following logic is applied.

1. A message of the required size, priority, routing class, transport protocol, and packetizing delay is created.
2. The message text is set.
3. The destination is set. This must be the node which sent the message which triggered the application.
4. If required outstanding ACK packets from the flow control windowing have been received, a packet is created. The node is made busy by the packetizing delay. If an ACK is needed then the packet creation attempt will be attempted on receipt of the ACK. Note that the first packet will never wait for an ACK packet—only subsequent packets may have to wait.
5. First the node is made busy by the packet switching delay defined for the node. Then a packet processing delay occurs, and finally a routing decision is made to determine which buffer to send the packet to.
6. If no space is available in the output buffer, the packet is dropped if the transport protocol does not specify retries. If retries are specified the packet is dropped, the retry time elapses, and then the packet is re-created and reattempted again from step 4.

Steps 4 through 6 are repeated for all packets in the answer message.

At the node there may be more than one application scheduled and in the process of generating the required packets for the respective messages. This also includes message sources, session sources and response sources as well as commands called directly from applications. *In fact the respective sources map onto an application which has a command list containing either a single transport command, setup command, or answer command respectively.*

In this case, a list of applications that are concurrently scheduled to generate packets at the node is kept. The application at the top of the list is selected and a packet for its message created as outlined above. Application scheduling is not quite round robin: an application waiting for flow control authorization will not block another application, even if it's at the top of the application queue. After execution the application is then placed at the bottom of the list. When the node becomes free, a packet for the next application is created.

### **Reporting:**

*See the following reports:*

*Message Delays For Transport And Answer Commands*

*Packet Statistics For Transport And Answer Commands*

*Received Message Counts*

Also individual node and link reports indicate loading by the packets generated from the Answer command.

### **Fields:**

<b>Name</b>	An alphanumeric field to identify the command. The name must be unique on the node, or unique in the global command list. If the name is changed, then any references to the name in an application command list will be changed automatically. The name is case sensitive.
<b>Priority</b>	Integer value ( <i>see Message: Priority</i> )
<b>Packet Routing Class</b>	Pick from list ( <i>see Routing Class: Packet</i> )
<b>Transport Protocol</b>	Pick from list ( <i>see Transport Protocol</i> )
<b>Packetizing Delay</b>	Real value including distributions ( <i>see Message: Packetizing Delay</i> )
<b>Message Size Units</b>	Bytes or Packets ( <i>see Message: Size Units</i> )
<b>Message Size Calculation</b>	Distribution Message Linear File Linear ( <i>see Message: Size Calculation</i> )
<b>Message Text</b>	Use Original Message Copy Message Name Set Message Text ( <i>see Message: Text</i> )

## 4. Modeling Constructs

### 4.8 Command: Process Data

#### *Purpose:*

A Process Data command is used within an application to model some processing task on a node. This would be used to represent some major subroutine or software task that has to run on the node. The loading is effectively defined in time duration terms as the node has a cycle time and the processing load is described as the number of cycles to execute.

#### *Creating:*

Processing commands can be created in one of three ways.

Firstly, they may be created locally on computer & communication nodes, and computer group nodes, and router nodes. The respective node dialog box has a **Commands** button which, when pressed, brings up a dialog box for editing and creating commands. If this method is used then the respective processing command has local scope to the node on which it is defined.

Secondly, Process Data commands may also be created using the **Define/Commands** menu option. In this case, the scope of the command is global to the model. A command defined globally can be executed by any node in response to an application which runs on the node and calls the global command.

Thirdly, Process Data commands can be added to the library via the **Archive/Object** menu option and then picking the **Commands** line from the offered list box. In this case the new processing command is saved in a library and is then available for use across models. To use a Process Data command defined this way in a particular model, it must be copied out of the library file to the model by using either the first or second method outlined above. It will then be either local to a node or global, depending on the method used.

#### *Connectivity:*

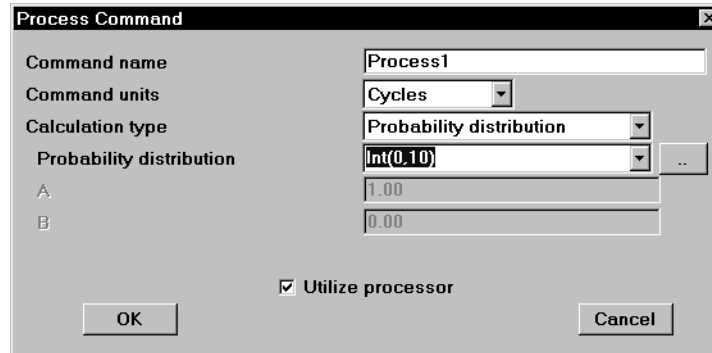
A Process Data command must be defined on a computer & communications node, on a computer group node, or as a global command from the **Define/Commands** menu. Once defined, it may be referenced in the command list of any application which runs on the node.

If a global command and a local command have the same name, then the local command will take precedence.

#### *Editing:*

Edit the details of the Process Data command from the node dialog boxes, or from the **Define/Commands** menu option, or from the **Archive/Objects** menu option.

When the processing command is edited in the library (via **Archive/Objects**) the changes made are not automatically copied into the local model, or other models that use the library defined command. For the changes to take effect the processing command must be recopied from the library.



**Processing Command Dialog Box**

**Execution:**

A Process Data command is executed by being called from an application which has been scheduled on a node. The application requests its commands to be executed one after the other. When a processing command is picked, the time duration of processing is calculated (see Number of Cycles below). Two scenarios are then possible depending upon whether the node has a time slice defined or not.

**No Time Slicing**

The node is made busy for the duration of the processing command. The application then continues to the next command in its command list.

**With Time Slicing**

Assuming the processing command duration is longer than 1 time slice, the node is made busy for the duration of a time slice and the amount of time remaining to complete the processing command is reduced by one time slice. The application is then interrupted and put at the bottom of the pending application list and another application resumed. Eventually, on a round robin basis, the interrupted application is resumed and another time slice worth of processing incurred. This process is repeated until all of the processing time has completed. (Note: It is likely that the last execution will not require a full time slice to complete). The application then continues to the next command in its command list.

**Reporting:**

*See the following reports:*  
*Node Utilization*  
*Application Delay*

**Fields:**

**Name**

An alphanumeric field to identify the command. The name must be unique on the node. If the name is changed, then any references to the name in an application command list will also be changed. The name is case sensitive.

**Number Of Cycles**

The number of cycles to execute when the processing command is invoked. The time that this will take is computed as the number of cycles multiplied by the node parameter: processing time per cycle. The number of cycles can be specified in the following terms:

**Probability Distribution** (*see Distributions*)

Any of the available COMNET III distributions.

#### 4. Modeling Constructs

##### **Based On Message Size**

If the application which wants to execute the processing command was itself scheduled by received message, the size of the scheduling message (in bytes) can be used to calculate the number of cycles. This is done on a linear calculation where a multiplier and an offset can be specified. The size calculation is then of the form:

$$\text{Multiplier} * \text{Message Size} + \text{Offset}$$

##### **Based On File Size**

If the application which wants to execute the processing command has read a file, the size of the first file read can be used to calculate the size of the message. This is done on a linear calculation where a multiplier and an offset can be specified. The size calculation is then of the form:

$$\text{Multiplier} * \text{Number of Bytes Read} + \text{Offset}$$

## 4.9 Command: Read File

### *Purpose:*

A Read File command is used within an application to model the time delay associated with reading from a local disk. The disk access times are specified on the node, and the amount of information to read is specified in the read file command. The time delay to execute the read can then be computed.

In addition to modeling workload on a node, the read may be used to obtain the number of bytes stored by an early (possibly asynchronous) write, and COMNET III monitors reads to count errors where more bytes read than are available in the file.

### *Creating:*

Read File commands can be created in one of three ways.

Firstly, they may be created locally on computer & communication nodes, and computer group nodes and router nodes. The respective node dialog box has a Commands button which, when pressed, brings up a dialog box for editing and creating commands. If this method is used then the respective read file command has local scope to the node on which it is defined.

Secondly, read file commands may also be created using the Define/Commands menu option. In this case, the scope of the command is global to the model. A command defined globally can be executed by any node in response to an application which runs on the node and calls the global command. A read file command defined in this way may only access the file called GENERAL STORAGE.

Thirdly, read file commands can be added to the library via the Archive/Object menu option and then picking the Commands line from the offered list box. In this case the new read file command is saved in a library and is then available for use across models. To use a read file command defined this way in a particular model then it must be copied out of the library file to the model by using either the first or second method outlined above. It will then be either local to a node or global, depending on the method used.

### *Connectivity:*

A Read File command must be defined on a computer & communications node, on a computer group node, on a router node, or as a global command from the Define/Commands menu. Once defined, it may be referenced in the command list of any application which runs on the node.

If a global command and a local command have the same name, then the local command will take precedence.

### *Editing:*

Edit the details of the Read File command from the node dialog boxes, or from the Define/Commands menu option, or from the Archive/Objects menu option.

When the Read File command is edited in the library (via Archive/Objects) the changes made are not automatically copied into the local model, or other models that use the library defined command. For the changes to take effect the Read File command must be recopied from the library.

## 4. Modeling Constructs



**Read Command Dialog Box**

### *Execution:*

A Read File command is executed by being called from an application which has been scheduled on a node. The application requests its commands to be executed one after the other. When a Read File command is picked, the file to access and the number of bytes to read is established. (*see File To Access and Bytes To Read Calculation below*). The file is then read using the following logic:

#### **1. Lock The File**

The file is locked for exclusive access by this read command. With no time slicing, locking the file has little impact—the node must perform the Read File command on the particular file to completion and so another Read Command could not try to simultaneously access the file anyway.

With time slicing, more than one application may attempt to access the file simultaneously for either reading or writing. The first application to call its file access command will lock the file and its command will then complete over one or more time slices. When finished, it will unlock the file and then the other application will commence its access.

#### **2. Calculate Time Duration**

The time duration needed to read the required number of bytes is calculated using the algorithm:

Duration = Number Of Sectors (Transfer Time + Seek Time)

where

Number Of Sectors = (Bytes To Read / Sector Size) rounded up

Transfer Time is specified as a node parameter

Seek Time is specified as a node parameter



### 3. Busy The Processor

The node's processor is made busy for the duration of the Read File command. Without time slicing, this is a straightforward time delay.

With time slicing, the node is made busy for as many time slices as are required to complete the calculated time duration. The last time slice may be a partial time slice.

When one time slice completes the application is interrupted and placed at the bottom of the pending application list on the node. The top application on the list restarted. The restarted application resumes its current command and has the node for one time slice. Eventually the interrupted application is resumed and so uses another time slice to count towards the read duration. Between time slices the file being read remains locked. Therefore other applications which may want to access the file cannot do so and will not consume time slices until the first application has read the file and unlocked it so that the file is then available.

### Update File Information

The respective file information is updated to reflect its new size after the read command. The file is then unlocked. If time slicing operation is being used the file is now available for other read or write accesses by other applications. The Read File command is now complete.

#### *Reporting:*

See the following reports:

Node Utilization  
Application Delay  
File Warnings

#### *Fields:*

<b>Name</b>	An alphanumeric field to identify the command. The name must be unique on the node. If the name is changed, then any references to the name in an application command list will also be changed. The name is case sensitive.
<b>File To Access</b>	The name of the file to read from. If the file does not exist on the local disk at the time of executing the command a run time warning will be put on the File Warning report and the command will continue as if the file exists—or at least the time delay will be incurred but the file will not be created. This is taken as the pessimistic response to this error.
<b>File Modification Method</b>	As data is to be read from the file, you may specify the effect on the file. The allowed choices are: <ul style="list-style-type: none"> <li><b>Do Not Modify</b> The file size is left untouched.</li> <li><b>Decrement File</b> The file is decremented by the number of bytes read. If more bytes are requested than available a warning is put on the File Warning report but the command continues as if all the requested bytes were read.</li> <li><b>Delete File</b> After reading the requested bytes the file is erased from the local disk on the</li> </ul>

## 4. Modeling Constructs

node.

### Bytes To Read Calculation

The number of bytes to read from the file must be specified. If more bytes are requested than available a warning is put on the File Warning report and the command continues as if all the requested bytes were read. The calculation choices include:

#### **Entire File**

Read however many bytes are currently in the file.

#### **Probability Distribution** (*see Distributions*)

Any of the available COMNET III distributions.

#### **Based On Message Size**

If the application which wants to execute the read file command was itself scheduled by received message, the size of the scheduling message (in bytes) can be used to calculate the number of bytes to read. This is done with a linear calculation where a multiplier and an offset can be specified. The size calculation is then of the form:

$$\text{Multiplier} * \text{Message Size} + \text{Offset}$$

#### **Based On File Size**

If the application which wants to execute the read file command has previously read a file, the size of the first read can be used to calculate the size of the subsequent read. This is done with a linear calculation where a multiplier and an offset can be specified. The size calculation is then of the form:

$$\text{Multiplier} * \text{First Number of Bytes Read} + \text{Offset}$$

## 4.10 Command: Setup Session

### *Purpose:*

A Setup Session command is used within an application to establish a session between the node and some other node in the network. This is useful for modeling switched or permanent virtual circuits which arise from within applications.

When a setup session command is executed a session setup packet is created and routed to the destination. A session confirmation packet is then returned to the origin via the same route (assuming connection-oriented routing is specified). Messages are then created on an interarrival pattern, each of which results in packets being transmitted to the destination over the session route. When all of the packets for all of the messages in the session have been received the session is complete.

Responses may be triggered at the destination by the session messages. In this case the response packets will follow the session route back to the origin (assuming connection-oriented routing). The session will not be complete until all the message responses have completed.

### *Creating:*

Setup Session commands can be created in one of three ways.

Firstly, they may be created locally on computer & communication nodes, router nodes, and computer group nodes. The respective node dialog box has a **Commands** button which, when pressed, brings up a dialog box for editing and creating commands. If this method is used then the respective Setup Session command has local scope to the node on which it is defined.

Secondly, Setup Session commands may also be created using the **Define/Commands** menu option. In this case, the scope of the command is global to the model. A command defined globally can be executed by any node in response to an application which runs on the node and calls the global command.

Thirdly, Setup Session commands can be added to the library via the **Archive/Object** menu option and then picking the **Commands** line from the offered list box. In this case the new Setup Session command is saved in a library and is then available for use across models. To use a Setup Session command defined this way in a particular model, it must be copied out of the library file to the model by using either the first or second method outlined above. It will then be either local to a node or global, depending on the method used.

### *Connectivity:*

A Setup Session command must be defined on a computer & communications node, on a computer group node, on a router node, or as a global command from the **Define/Commands** menu. Once defined, it may be referenced in the command list of any application which runs on the node.

If a global command and a local command have the same name, then the local command will take precedence.

### *Editing:*

Edit the details of the Setup Session command from the node dialog boxes, or from the **Define/Commands** menu option, or from the **Archive/Objects** menu option.

When the Setup Session command is edited in the library (via **Archive/Objects**) the changes made are not automatically copied into the local model, or other

## 4. Modeling Constructs

models that use the library defined command. For the changes to take effect the Setup Session command must be recopied from the library.

Name	Setup1	Priority	1
Routing class	Standard	Transport protocol	Generic
Packetizing (ms)	0.0	Net svc level	1
Messages per session	1.0	Setup packet (bytes)	5
Msg interarrival (sec)	Exp(10.0)	Confirm packet (bytes)	5
Message size	1000.0	Msg size units	Bytes
Msg text option	Copy message name	Dest type	Random neighbor
Msg text			

**Setup Session Dialog Box**

### *Execution:*

A Setup Session command is executed by being called from an application which has been scheduled on a node. The application requests its commands to be executed one after the other. When it is the turn of a Setup Session command to execute the following logic is applied.

1. A session is created. The destination is selected from the user defined choice.
2. A session setup packet is created. A setup command finishes executing as soon as the setup packet is created. Once the session has been established (after the return of the connect packet), COMNET III spawns transport commands as required to produce messages in the session.

The packetizing delay is not incurred for session setup packets. The setup packet is delayed by the Processing Delay For Session Setup Packets as defined on the node. A routing decision is then taken based on the current routing algorithm and the session setup packet put into the chosen output buffer.

If the routing algorithm does not find a route the session setup packet is blocked. If the packet is originating in a subnet and destined for the backbone or another subnet, the routing decision is to reach an access point. The packet may therefore reach the selected access point, at which time another routing decision will be made. This second routing decision may also return no available route. Therefore the session setup packet may be blocked at the access point, or any node along the way, as well as at the origin.

The selected output buffer may have insufficient space to accept the session setup packet. Again the session setup packet is blocked.

Another source of blocking for session setup packets is the session limit for nodes and links (in subnets that use connection-oriented routing).

When a block occurs and if the routing class for the session specifies a session retry interval, the session setup packet will be recreated and reattempted after the specified retry time. The attempt will continue to be made until a route or buffer space is found or the simulation terminates. If no session retry interval is specified (i.e. set to None) the session is dropped and the application moves to its next command.

The session setup packet is then routed to the destination. At each node the Processing Delay For Session Setup Packets is incurred, plus any buffer switching or queueing delays. If the session setup packet is blocked at any intermediate point then again retries will be attempted from the origin depending on the routing class settings.

3. Eventually the session setup packet reaches the destination node. It is received through the input buffer and then incurs the Processing Delay For Session Setup packets on the destination node. At this point a session confirm packet is created and routed back to the originating node, reversing the route traversed by the setup packet but undergoing no further processing delay for Session Setup Packets.
4. On receipt of the session confirm packet at the originating node, a message of the required size, priority, routing class, transport protocol, and packetizing delay is created.
5. The message text is set.
6. The destination is set. This is the same as the destination picked for the session setup packet.
7. If required outstanding ACK packets from the flow control windowing have been received, a packet is created. The node is made busy by the packetizing delay. If an ACK is needed then the packet creation attempt will be attempted on receipt of the ACK. Note that the first packet will never wait for an ACK packet—only subsequent packets may have to wait.
8. A routing decision is made to determine which buffer to send the packet to. If connection oriented routing for sessions is selected on the routing protocol then the message packet will follow the same route as the session setup packet. The node is made busy by the packet switching delay defined for the node.
9. If no space is available in the output buffer, the packet is dropped if the transport protocol does not specify retries. If retries are specified the packet is dropped, the retry time elapses, and then the packet is recreated and reattempted again from step 4.

Steps 7 through 9 are repeated for all packets in the message.

If there is more than one message specified then, after the requisite interarrival time, another message is created and steps 5 through 9 are repeated for this next message.

#### ***Scheduling:***

At the node there may be more than one application scheduled and in the process of generating the required packets for the respective messages. This also includes message sources, session sources and response sources as well as commands called directly from applications. In fact the respective sources map onto

## 4. Modeling Constructs

an application which has a command list containing either a single transport command, session setup command, or answer message command respectively.

In this case, a list of applications that are concurrently scheduled to generate packets at the node is kept. The application at the top of the list is selected and a packet for its message created as outlined above. Application scheduling is not quite round robin: an application waiting for flow control authorization will not block another application, even if it's at the top of the application queue. After execution the application is then placed at the bottom of the list. When the node becomes free, a packet for the next application is created.

### **Reporting:**

Messages: Setup Commands  
Packets: Delays For Setup Commands  
Sessions: Setup Delay By Setup Command  
Sessions: Length by Session Command  
Sessions: Blocking by Session Command

Node: Utilization (Loaded by session packets)  
Node: Application Delays (Session command is part of delay)  
Link: Delays and Utilization (Loaded by session packets)  
Link: Random Access Link Performance (Loaded by session packets)

Sessions: By Node (For connection-oriented routing only)  
Sessions: By Link (For connection-oriented routing only)

### **Fields:**

<b>Name</b>	An alphanumeric field to identify the command. The name must be unique on the node. If the name is changed, then any references to the name in an application command list will also be changed automatically. The name is case sensitive.
<b>Priority</b>	Integer value ( <i>see Packet: Priority</i> )
<b>Packet Routing Class</b>	Pick from list ( <i>see Routing Class: Packet</i> )
<b>Transport Protocol</b>	Pick from list ( <i>see Transport Protocol</i> )
<b>Setup Packet Size</b>	When a session command executes, a setup packet is created at the originating node and routed to the destination. The setup packet size is the size in bytes of this packet.
<b>Confirm Packet Size</b>	When the setup packet of the session arrives at the destination, a confirm packet is returned to the origin by the same route (assuming connection oriented routing for packets). The confirm packet size is the size in bytes of this packet.
<b>Messages Per Session</b>	Once the session has been established by the confirm packet arriving back at the origin, messages are then generated at the origin and sent to the destination. The number of messages to send may be specified—after all of these messages have arrived at the destination the session is complete.  The messages per session is entered as a statistical distribution. ( <i>see Distributions</i> )

To model a permanent virtual circuit established as part of an application, set the number of messages per session to a value which will take longer to complete than the simulation run length will allow. The session will then be up for the duration of the simulation. A setup command finishes executing immediately after the setup packet is created.

**Message Interarrival Time** The interarrival time between the messages of the session. The first message is sent immediately after the confirm packet is received; all other messages follow with specified interarrival time.  
(see *Message: Interarrival Time*)

**Message Size Units** Bytes or Packets  
(see *Message: Size Units*)

**Message Size Calculation** Distribution  
(see *Message: Size Calculation*)

Message Linear is not supported on sessions.

File Linear is not supported on sessions.

**Message Text** Copy Message Name  
Set Message Text  
(see *Message: Text*)

Use Original Message is not supported on sessions.

**Packetizing Time** The time it takes to make a packet for the messages on this session.  
(see *Packet: Packetizing Delay*)

**Destination Type** The destination of the messages on the session. The session setup packet is routed to the destination, the connect packet is returned to the origin, and then the session messages are sent to the destination a packet at a time.

The destination type may be picked from:

**Random List** (see *Destination: Random List*)

**Random Neighbor** (see *Destination: Random Neighbor*)

**Weighted List** (see *Destination: Weighted List*)

**Multicast List**(see *Destination: Multicast List*)

## 4. Modeling Constructs

### 4.11 Command: Transport Message

***Purpose:***

A Transport Command is used within an application to send a message between the node and some other node in the network. The inherent routing is on a data-gram basis where each packet is individually routed to the destination. However, the routing protocols in force may route each packet in the message over the same path.

***Creating:***

Transport commands can be created in one of three ways.

Firstly, they may be routed locally on computer & communication nodes and computer group nodes. The respective node dialog box has a Commands button which, when pressed, brings up a dialog box for editing and creating commands. If this method is used then the respective Transport command has local scope to the node on which it is defined.

Secondly, Transport commands may also be created using the Define/Commands menu option. In this case, the scope of the command is global to the model. A command defined globally can be executed by any node in response to an application which runs on the node and calls the global command.

Thirdly, Transport commands can be added to the library via the Archive/Object menu option and then picking the Commands line from the offered list box. In this case the new Transport command is saved in a library and is then available for use across models. To use a Transport command defined this way in a particular model, it must be copied out of the library file to the model by using either the first or second method outlined above. It will then be either local to a node or global, depending on the method used.

***Connectivity:***

A Transport command must be defined on a computer & communications node, on a computer group node, on a router node, or as a global command from the Define/Commands menu. Once defined, it may be referenced in the command list of any application which runs on the node.

If a global command and a local command have the same name, then the local command will take precedence.

***Editing:***

Edit the details of the Transport command from the node dialog boxes, or from the Define/Commands menu option, or from the Archive/Objects menu option.

When the Transport command is edited in the library (via Archive/Objects) the changes made are not automatically copied into the local model, or other models that use the library defined command. For the changes to take effect the Transport command must be recopied from the library.



The screenshot shows a dialog box titled "Transport Command". It contains the following fields and controls:

- Name:** Transport1
- Priority:** 1
- Routing class:** Standard
- Transport protocol:** Generic
- Packetize (ms):** 0.0
- Net svc level:** 1
- Msg size units:** Bytes
- Msg size calc:** Probability distribution
- Prob distrib:** 1000.0
- A:** 1.000
- B:** 0.000
- Msg text opt.:** Copy message name
- Msg text:** (empty)
- Dest type:** Random neighbor
- Buttons:** OK, Cancel, Edit Destination List...

Transport Command Dialog Box

**Execution:**

A Transport command is executed by being called from an application which has been scheduled on a node. The application requests its commands to be executed one after the other. When it is the turn of a Transport command to execute the following logic is applied.

1. A message of the required size, priority, routing class, transport protocol, and packetizing delay is created.
2. The message text is set.
3. The destination is set. This may be either a single node destination, or a multi-cast list where the message needs to be copied to several destinations.
4. If required outstanding ACK packets from the flow control windowing have been received, a packet is created. The node is made busy by the packetizing delay. If an ACK is needed then the packet creation attempt will be attempted on receipt of the ACK. Note that the first packet will never wait for an ACK packet—only subsequent packets may have to wait.
5. A routing decision is made to determine which buffer to send the packet to. The node is made busy by the packet switching delay defined for the node.

If the routing algorithm does not find a route the packet is blocked. It is optionally retried later depending on the retry characteristics defined under the transport protocol. If the packet is originating in a subnet and destined for the backbone or another subnet, there may be several routing decision before reaching the access point. Each decision brings the packet one node closer. The packet may reach the selected access point, at which time another routing decision will be made. This second routing decision may also return no available route. Therefore the packet may be blocked at the access point as well as at the origin.

6. If no space is available in the output buffer, the packet is dropped if the transport

## 4. Modeling Constructs

protocol does not specify retries (see Transport Protocol). If retries are specified the packet is dropped, the retry time elapses, and then the packet is recreated and reattempted again.

7. The packet then is transmitted across each link and through each node or access point until it reaches its destination. If a block occurs at any point then the packet is dropped and optionally retried from the origin depending upon the transport protocol settings.
8. If the destination of the message is a multicast list then a routing decision is made at the origin for all of the destinations in the list. This may result in several routes being picked, some of which share initial hops. Where an initial hop is common to several routes then the packet will only be transmitted once across the hop. At the node where the routes diverge a packet will be cloned for each outgoing hop as determined by the routing decision.
9. When the packet reaches its destination, and if flow control is in operation, and if the receipt of the packet demands that an ACK be sent back to the origin, then an ACK will be created at the destination and routed back to the origin using the same routing class, transport protocol and routing algorithms as the received packet.

Steps 4 through 9 are repeated for all packets in the message.

At the node there may be more than one application scheduled and in the process of generating the required packets for the respective messages.

This also includes message sources, session sources and response sources as well as commands called directly from applications. In fact the respective sources map onto an application which has a command list containing either a single transport command, session setup command, or answer message command respectively.

In this case, a list of applications that are concurrently scheduled to generate packets at the node is kept. The application at the top of the list is selected and a packet for its message created as outlined above. Application scheduling is not quite round robin: an application waiting for flow control authorization will not block another application, even if it's at the top of the application queue. After execution the application is then placed at the bottom of the list. When the node becomes free, a packet for the next application is created.

### ***Reporting:***

Messages: Delays For Transport & Answer Commands

Packets: Delays For Transport & Answer Commands

The following reports indicate loading caused by packets from Transport Commands:

Node Utilization

Node Application Delays

Node Received Messages

Link Delays and Utilization

Link Random Access Link Performance

Buffer Input By Node

Buffer Input By Port

Buffer Output By Node

Buffer Output By Port

**Fields:**

<b>Name</b>	An alphanumeric field to identify the command. The name must be unique in the node's command list. If the name is changed, then any references to the name in an application command list will also be changed. The name is case sensitive.
<b>Priority</b>	Integer value ( <i>see Packet: Priority</i> )
<b>Packet RoutingClass</b>	Pick from list ( <i>see Routing Class: Packet</i> )
<b>Transport Protocol</b>	Pick from list ( <i>see Transport Protocol</i> )
<b>Message Size Units</b>	Bytes or Packets ( <i>see Message: Size Units</i> )
<b>Message Size Calculation</b>	Distribution Message Linear. File Linear. ( <i>see Message: Size Calculation</i> )
<b>Message Text</b>	Copy Message Name Set Message Text Use Original Message ( <i>see Message: Text</i> )
<b>Packetizing Time</b>	The time it takes to make a packet for the messages on this transport command. ( <i>see Packet: Packetizing Delay</i> )
<b>Destination Type</b>	The destination of the messages on the session. The session setup packet is routed to the destination, the connect packet is returned to the origin, and then the session messages are sent to the destination a packet at a time.  The destination type may be picked from: <b>Random List</b> ( <i>see Destination: Random List</i> ) <b>Random Neighbor</b> ( <i>see Destination: Random Neighbor</i> ) <b>Weighted List</b> ( <i>see Destination: Weighted List</i> ) <b>Multicast List</b> ( <i>see Destination: Multicast List</i> )

## 4. Modeling Constructs

### 4.12 Command: Wait For

*Purpose:*

The Wait For command evaluates a condition, and if the condition is met, it completes and gets out of the way for the next command to run. If the condition is not met, Wait For blocks until the condition is met, or until the time-out timer expires, whichever comes first. By “block,” we mean the application running the command sequence is suspended.

The condition can be of three types: **Received Message**, **Alarm**, or **Expression**.

**Wait For Received Message:** Causes an application to suspend execution until the received message text requirement for the Wait For command is satisfied. The only attribute of a Wait For command besides its name is its received message text requirement. When a message arrives at a node, any suspended applications get to use the incoming message text before any new applications are scheduled. There are three additional options that may be set by checking "Get required msg text from command's input parameter", "Retain previous received message (if any)", and "Only accept msg responding to this app" options.

The "Get required msg text from command's input parameter" allows the Wait For command to utilize an output parameter generated by a previous command in the command sequence, as a response mechanism to complete the Wait For command.

The "Retain previous received message (if any)" option is useful for calculating the round-trip time for a Wait For message. To better illustrate this, assume that a node sends a message request to a server requesting a database file located on another host system. In this scenario, the server is the connecting link between the workstation and the host (a typical client/server application). The application source on the file server responds to the message request and generates a message request to the host system, requesting the database file. The application source then waits for the database file to be returned. When the database file returns to the server, the application source answers the request using the original workstation message request, and sends the file to the workstation. It is important to note that when the application source sends the file to the workstation, the message text option needs to be set to "use original message" in order to calculate and track the total round-trip time for the original workstation request.

The "Only accept msg responding to this app" option restricts the Wait For command to respond to messages only from specific application instances that were triggered by earlier messages from the same application that has this Wait For command. The purpose of this switch is to bind two application instances to interact in a sequential way.

In modeling detailed interactions between two nodes (client-server, for instance) there may be a pair of detailed applications, each with a long sequence of commands separated by Wait For commands that wait for a message from the other application. This is fairly simple to model as long as there is just one instance of both applications. The new option is required when one side (for example, the server side) has multiple instances of the application conversing with several clients. In this case, there may be several instances of the application waiting for the same message text but from different application instances -- a message that arrives at this node may end up triggering a Wait For command in a different application instance than intended. The new option allows the application

instance to respond to only messages from a specific application instance.

The option has a couple drawbacks that can cause the application to never complete. In one case, the Wait For command can activate too late and miss the required message. In another case, the required message may not get delivered because the message experiences errors and the transport protocol does not retransmit blocked packets. Please see the **Stop Waiting** section below for a solution to this problem.

The Wait For command is especially useful for modeling client-server operations with dependent sequences of messages going back and forth between client and server. Without this command, it would be necessary to have a separate application source for each type of incoming message that requires a different response. By using the “Wait For” command, a single application can send a message, suspend until it gets a response, send another message and then suspend again until it gets the appropriate response, etc. The Wait For command makes it possible to build process-oriented models of applications, in contrast to event-oriented models.

**Wait For Alarm:** If the specified alarm is already sounding when this command executes, no blocking occurs, the command completes and the next command is executed. If the alarm is not already sounding, the command blocks until the alarm does sound.

The alarm combo box shows all the alarms that are currently set. Use the two-dot button to set additional alarms if necessary, and pick an alarm. Then specify the node or link on which the alarm must sound.

**Wait For Expression:** The expression can be almost any expression, involving constants, user variables, and most of the intrinsic functions. Some intrinsic functions, like SimTime, are prohibited. The expression must evaluate to a boolean value.

When the command is executed, the expression is evaluated, and if it is TRUE, no blocking occurs, the command completes and the next command is executed. If the expression is FALSE, the command blocks until the expression becomes TRUE. Of the limited number of things permitted in a Wait For expression, only user variables are liable to change after the expression is evaluated the first time. Thus, the expression is only re-evaluated if and when referenced variables change.

Clearly, a Wait For expression that does not involve one or more variables is not of much use, UNLESS the “stop waiting” option is checked. If the “stop waiting” option is checked, then either the expression is FALSE when first evaluated, or the command will block for the duration of the time-out time.

**Stop Waiting:** The blocking process you’re modeling will probably not wait forever. To model a process that times out, check the “stop waiting after” option, and give the time limit you’re willing to wait. After that time limit, the command and the application will be terminated. To catch this event and do something special in response to it, click the Triggered Sources button on the application source, select the source that will perform the special response, and use the “upon wait timeout” triggering rule.

## 4. Modeling Constructs

**Troubleshooting:** With the increased flexibility of the Wait For command comes increased complexity and potential for mistakes in the model. To assist in finding wayward applications that are in perpetual Waits, turn on the Suspended/Pending Apps snapshot. This snapshot will show you a count of suspended and pending applications on any node on which it's enabled.

### ***Creating:***

Wait For commands can be created in one of three ways.

Firstly, they may be created locally on computer & communication nodes, and computer group nodes, and router nodes. The respective node dialog box has a Commands button which, when pressed, brings up a dialog box for editing and creating commands. If this method is used then the respective processing command has local scope to the node on which it is defined.

Secondly, Wait For commands may also be created using the Define/Commands menu option. In this case, the scope of the command is global to the model. A command defined globally can be executed by any node in response to an application which runs on the node and calls the global command.

Thirdly, Wait For commands can be added to the library via the Archive/Object menu option and then picking the Commands line from the offered list box. In this case the new processing command is saved in a library and is then available for use across models. To use a Wait For command defined this way in a particular model, it must be copied out of the library file to the model by using either the first or second method outlined above. It will then be either local to a node or global, depending on the method used.

### ***Connectivity:***

A Wait For command must be defined on a computer & communications node, on a computer group node, or as a global command from the Define/Commands menu. Once defined, it may be referenced in the command list of any application which runs on the node.

If a global command and a local command have the same name, then the local command will take precedence.

### ***Editing:***

Edit the details of the Wait For command from the node dialog boxes, or from the Define/Commands menu option, or from the Archive/Objects menu option.

When the Wait For command is edited in the library (via Archive/Objects) the changes made are not automatically copied into the local model, or other models that use the library defined command. For the changes to take effect the processing command must be recopied from the library.

### ***Reporting:***

*See the following reports:*  
*Node Utilization*  
*Application Delay*

### ***Fields:***

#### **Wait For Command Name**

An alphanumeric field to identify the command. The name must be unique on the node, or unique in the global command list. If the name is changed, then any references to the name in an application command list will be changed automatically. The name is case sensitive.

**Wait For Expression:** The expression can be almost any expression, involving constants, user variables, and most of the intrinsic functions. Some intrinsic functions, like SimTime, are prohibited. The expression must evaluate to a boolean value.

When the command is executed, the expression is evaluated, and if it is TRUE, no blocking occurs, the command completes and the next command is executed. If the expression is FALSE, the command blocks until the expression becomes TRUE. Of the limited number of things permitted in a Wait For expression, only user variables are liable to change after the expression is evaluated the first time. Thus, the expression is only re-evaluated if and when referenced variables change.

Clearly, a Wait For expression that does not involve one or more variables is not of much use, UNLESS the “stop waiting” option is checked. If the “stop waiting” option is checked, then either the expression is FALSE when first evaluated, or the command will block for the duration of the time-out time.

**Wait For Alarm** If the specified alarm is already sounding when this command executes, no blocking occurs, the command completes and the next command is executed. If the alarm is not already sounding, the command blocks until the alarm does sound.

The alarm combo box shows all the alarms that are currently set. Use the two-dot button to set additional alarms if necessary, and pick an alarm. Then specify the node or link on which the alarm must sound.

**Required Message Text** The message text that the Wait For Command is waiting for in order to continue the suspended application source.

**Get required msg text from command's input parameter**

This option allows the Wait For command to utilize an output parameter generated by a previous command in the command sequence, as a response mechanism to complete the Wait For command.

**Retain Previous Received Message If Any**

Used for calculating the round-trip time for a filtered message. Please see above section under Command:Wait For, for a full description.

**Only Accept Message Responding To This App**

This option restricts the filter command to respond to messages only from specific application instances that were triggered by earlier messages from the same application that has this filter command. The purpose of this switch is to bind two application instances to interact in a sequential way.

**Stop Waiting (and terminate app) after**

The blocking process you're modeling will probably not wait forever. To model a process that times out, check the “stop waiting after” option, and give the time limit you're willing to wait. After that time limit, the command and the application will be terminated. To catch this event and do something special in response to it, click the Triggered Sources button on the application source, select the source that will perform the special response, and use the “upon wait timeout” triggering rule.

## 4. Modeling Constructs

### 4.13 Command: Write File

#### *Purpose:*

A Write File command is used within an application to model the time delay associated with writing to a local disk. The disk access times are specified on the node, and the amount of information to be written is specified in the Write File command. The time delay to execute the Write can then be computed.

In addition to introducing workload, the Write command may be used to store a number of bytes for later access by the Read command. COMNET III will also flag errors when write exceeds the size of available storage.

#### *Creating:*

Write File commands can be created in one of three ways.

Firstly, they may be created locally on computer & communication nodes, router nodes, and computer group nodes. The respective node dialog box has a Commands button which, when pressed, brings up a dialog box for editing and creating commands. If this method is used then the respective Write File command has local scope to the node on which it is defined.

Secondly, Write File commands may also be created using the Define/Commands menu option. In this case, the scope of the command is global to the model. A command defined globally can be executed by any node in response to an application which runs on the node and calls the global command. A Write File command define in this way may only access the file called GENERAL STORAGE.

Thirdly, Write File commands can be added to the library via the Archive/Object menu option and then picking the Commands line from the offered list box. In this case the new Write File command is saved in a library and is then available for use across models. To use a Write File command defined this way in a particular model then it must be copied out of the library file to the model by using either the first or second method outlined above. It will then be either local to a node or global, depending on the method used.

#### *Connectivity:*

A Write File command must be defined on a computer & communications node, on a router node, on a computer group node, or as a global command from the Define/Commands menu. Once defined, it may be referenced in the command list of any application which runs on the node.

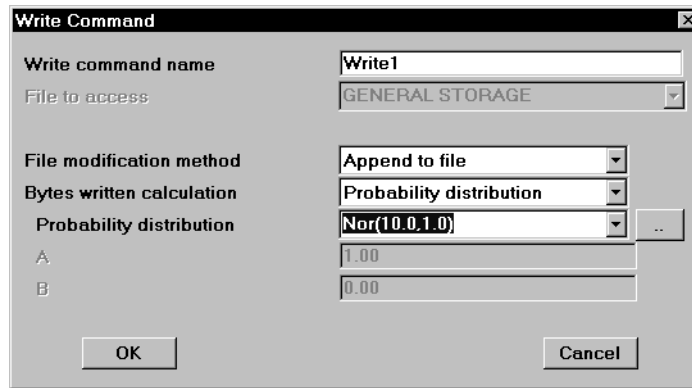
If a global command and a local command have the same name, then the local command will take precedence.

#### *Editing:*

Edit the details of the Write File command from the node dialog boxes, or from the Define/Commands menu option, or from the Archive/Objects menu option.

When the Write File command is edited in the library (via Archive/Objects) the changes made are not automatically copied into the local model, or other models that use the library defined command. For the changes to take effect the Write File command must be recopied from the library.





Write File Command Dialog Box

**Execution:**

A Write File command is executed by being called from an application which has been scheduled on a node. The application requests its commands to be executed one after the other. When a Write File command is picked, the file to access and the number of bytes to write is established. (see *File To Access and Bytes To Write Calculation* below). The file is then written using the following logic:

**1. Lock The File**

The file is locked for exclusive access by this Write File command. With no time slicing, locking the file has little impact—the node must perform the Write File command on the particular file to completion and so another Write File Command could not try to simultaneously access the file anyway.

With time slicing, more than one application may attempt to access the file simultaneously for either reading or writing. The first application to call its file access command will lock the file and its command will then complete over one or more time slices. When finished, it will unlock the file and then the other application will commence its access.

**2. Calculate Time Duration**

The time duration needed to write the required number of bytes is calculated using the algorithm:

$$\text{Duration} = \text{Number Of Sectors} (\text{Transfer Time} + \text{Seek Time})$$

where

$$\text{Number Of Sectors} = (\text{Bytes To Write} / \text{Sector Size}) \text{ rounded up}$$

Transfer Time is specified as a node parameter

Seek Time is specified as a node parameter

**3. Busy The Node**

The node is made busy for the duration of the Write File command. Without time slicing, this is a straightforward time delay.

With time slicing, the node is made busy for as many time slices as are required to complete the calculated time duration. The last time slice may be a partial time slice.

When one time slice completes and there are other applications pending, the exe-

## 4. Modeling Constructs

cutting application is interrupted and placed at the bottom of the pending application list on the node. The top application on the list restarted. The restarted application resumes its current command and has the node for one time slice. Eventually the interrupted application is resumed and so uses another time slice to count towards the write duration. Between time slices the file being written remains locked. Therefore other applications which may want to access the file cannot do so and will not consume time slices until the first application has written the file and unlocked it so that the file is then available.

### **Update File Information**

The respective file information is updated to reflect its new size after the Write File command. The file is then unlocked. If time slicing operation is being used the file is now available for other access by other applications. The Write File command is now complete.

### **Reporting:**

See the following reports:  
Node Utilization  
Application Delay  
File Warnings

### **Fields:**

#### **Name**

An alphanumeric field to identify the command. The name must be unique on the node. If the name is changed, then any references to the name in an application command list will also be changed. The name is case sensitive.

#### **File To Access**

The name of the file to write to. If the file does not exist on the local disk at the time of executing the command it will be created.

#### **File Modification Method**

As data is written to the file, you may specify the effect on the file. The allowed choices are:

##### **Append**

The file size is increased by the amount of data written. If the modeled disk drive overflows then a warning is written to the File Warnings report and the transfer takes the length of time needed to write all the data. The local disk is not increased in size.

##### **Replace**

The file is replaced by the number of bytes written. If the modeled disk drive overflows then a warning is written to the File Warnings report and the transfer takes the length of time needed to write all the data. The local disk is not increased in size.

##### **Update**

The file size is not changed but the time delay for writing is incurred. This models updating data in the middle of a file.

#### **Bytes To Write Calculation**

The number of bytes to write to the file must be specified. If more bytes are written than there is space available on the disk a warning is put on the File Warning report and the command continues as if all the requested bytes were written. The calculation choices include:

**Probability Distribution** (*see Distributions*)

Any of the allowed COMNET III distributions.

**Based On Message Size**

If the application which wants to execute the write file command was itself scheduled by received message, the size of the scheduling message (in bytes or packets) can be used to calculate the number of bytes to write. This is done on a linear calculation where a multiplier and an offset can be specified. The size calculation is then of the form:

$$\text{Multiplier} * \text{Message Size} + \text{Offset}$$

**Based On File Size**

If the application which wants to execute the write file command has previously read a file, the size of the first read can be used to calculate the size of the subsequent write. This is done with a linear calculation where a multiplier and an offset can be specified. The size calculation is then of the form:

$$\text{Multiplier} * \text{First Number of Bytes Read} + \text{Offset}$$

## 4. Modeling Constructs

### 4.14 COMNET Baseline

*Purpose:*

COMNET III provides an interface to a utility called COMNET Baseline that is used to import network topology files and traffic information gathered from network management discovery tools, and captured network traffic data from network analyzers and RMON and RMON II probes. Using COMNET Baseline, COMNET III can import topology data from the following tools:

Network General's - Expert Sniffer

Hewlett Packard's - OpenView

IBM's - Netview 6000

Cabletron Systems - SPECTRUM

Castlerock Computing - SNMPC

Digital Equipment Corporation's - Polycenter

Network traffic data can be imported from the following tools:

Network General's - Expert Sniffer, and Sniffer

Axon Networks' - LANServant

Frontier Systems' - NETscout

Hewlett Packard's - Netmetrix

Wandel and Golterman's - Domino analyzer

*For more information about AutoBaseline, see Chapter 11. COMNET Baseline and also the COMNET Baseline User's Guide.*

## 4.15 Destination: Least Busy List

<b><i>Purpose:</i></b>	From the list of destinations, the one whose processor is least busy is chosen. The statistic used to determine busyness is the same statistic used by the Processor Utilization snapshot report.
<b><i>Creating:</i></b>	Destination lists are created by clicking on the destination list button for the source which requires a destination. A list box then appears with options to add or remove nodes from the displayed destination list. For a Least Busy List the probability button is grayed out as the probability of sending to each destination in the list is governed by which destination is the least busy.
<b><i>Connectivity:</i></b>	<p>A Least Busy List is allowed as a destination choice for all types of message generators, other than a Response Source and an Answer Message Command. For these latter two types of generators the destination is implicitly the sender of an inbound message.</p> <p>A destination is any other node in the model, including nodes in the local subnetwork, remote subnetworks or the backbone. Subnetwork node names are prefixed with the subnetwork name. Destination nodes cannot include routers or ATM switches. Messages may be sent to SELF (i.e., the originating node). In this case the node will be busy by the packetizing delay and packet switching times for the packets in the message, but the packet will not be placed in an output port buffer as it is already at its destination.</p> <p>If a traffic generator should only send to a single destination, this may be entered as a Least Busy List with a single entry.</p>
<b><i>Editing:</i></b>	Edit the details of the destination list from the Destination List dialog box. You have the options to add a new destination to the list, or remove an existing destination from the list. The Set Probability button does not appear on a Least Busy Destination list because the probability of sending to each destination is determined by which destination is the least busy.
<b><i>Execution:</i></b>	<p>When a message, session or call is scheduled at some originating node, a destination has to be selected. If a Least Busy List is in use then the destination whose processor is being used the least is chosen from the list of destinations.</p> <p>When the least busy destination turns out to be a group node, or if the only member of the list is a group node, the specific destination in the group will be chosen by the same Least Busy algorithm.</p> <p>If two or more destinations are equally busy (for example, both have 0% utilization), the tie is broken by round robin selection. If all the nodes remain un-busy, the result will be identical to a round robin list.</p>
<b><i>Reporting:</i></b>	Node Received Message Counts.
<b><i>Fields:</i></b>	
<b>Destination Node Name</b>	The names of the destination nodes as defined on their dialog boxes. COMNET III makes references to the actual nodes, so that if the node's name changes, its name in the list will be changed automatically.

## 4. Modeling Constructs

### 4.16 Destination: Multicast List

**Purpose**

A multicast list destination is a list of destination names, all of which have a copy of the message sent to them.

When a message is to be broadcast to a multicast list, a route to each destination is established. Only one packet is transmitted over common links on routes to different destinations—at the point where the route splits then new packets are created to follow the different routes.

**Creating:**

Destination lists are created by clicking on the destination list button for the source which requires a destination. A list box then appears with options to add or remove nodes from the displayed destination list. For a multicast list the probability button is grayed out as the probability of sending to each destination in the list is 1.

**Connectivity:**

Multicast destinations are allowed for Message Sources and Transport Commands. Session Sources, Response Sources, Call Sources, Session Setup Commands and Answer Message Commands, cannot have a multicast list for their destination.

A destination is any other node in the model, including nodes in the local subnetwork, remote subnetworks or the backbone. Subnetwork node names are prefixed with the subnetwork name. Destination nodes cannot include routers or ATM switches. Messages may be sent to SELF (i.e., the originating node). In this case the node will be busy by the packetizing delay and packet switching times for the packets in the message, but the packet will not be placed in an output port buffer as it is already at its destination. (Note: For a multicast list that includes SELF then clones of the original packet must be made and sent on for the other destinations).

**Editing:**

Edit the details of the destination list from the Destination List dialog box. You have the options to add a new destination to the list, or remove an existing destination from the list. The Set Probability button does not appear on a multicast destination list because the probability of sending to each destination is automatically set to 1.



**Destination List Dialog Box**

**Execution:**

A multicast destination may be specified for a message source or a transport

command. The originating node executes the application which will generate the packets for the message that is to be sent. For a description of the execution logic see the appropriate section on transport commands, message sources, and node scheduling. The following differences should be noted:

At the originating node, and at intermediate nodes on route to one or more destinations, an output port has to be selected through which the packet may be forwarded to its destination(s). The routing algorithm determines whether multiple destinations may be reached by using either one port or by using several output ports. If more than one output port must be used then the packet is cloned the appropriate number of times and the cloned packets placed in the respective output port buffers. This cloning of the message packet occurs after any node processing delays immediately before placing the ongoing packet(s) into the output port buffer. Note that if multiple destinations can all be reached via the same output port then only one packet will leave the node and that packet will be placed in that output port.

No flow control is used when multicast destinations are specified, irrespective of the transport protocol settings. The sending application keeps generating packets as allowed by the node on which it is executing—no check is made that ACK packets have been received. Indeed, no ACK packets are generated at the destination node when packets addressed via a multicast list are received. The main reason for this is that one packet transmitted at the origin may be cloned several times on route to multiple destinations and so it is indeterminate how windowed flow control could work with several destinations.

If a packet which has a multicast destination is blocked, then it is counted as blocked and immediately dropped. No retries are scheduled irrespective of transport protocol settings. The main reason for this is that an original packet may have been cloned several times in order to reach multiple destinations. If one of the clones blocks then the original packet would have to be retransmitted, but it would have to then know which destinations not to send to (as other packets had already been sent to non-blocked destinations). While this may seem simple at face value, the problems of multiple blocking, subnetwork routing, link & node failures during retransmission attempts etc., make this a difficult implementation issue. Consequently, the simplification is that blocked packets sent to multicast destinations are dropped.

It is not recommended to use a multicast list for sending messages to a single destination. The previous three points mean that a message sent out to a single destination on a multicast list is treated differently from a message sent to a single destination on either a random list or a weighted list. For the multicast situation, even though there is only one destination, the flow control and retry characteristics are inoperative. Use a random list to send to a single destination.

***Reporting:***

Node Received Message Counts.

***Fields:***

**Destination Node Name**

The names of the destination nodes as defined on their dialog boxes. COMNET III makes references to the actual nodes, so that if the node's name changes, its name in the list will be changed automatically.

## 4. Modeling Constructs

### 4.17 Destination: Random List

#### *Purpose*

A random list destination is a list of destination names, one of which is picked at random when a destination is required. All destinations in the list have equal probability. Note that the probability assigned to a group node is weighted by the number in the group, so that all individual nodes used in the simulation have equal probability.

A random list is often used for sending messages to a single destination.

#### *Creating:*

Destination lists are created by clicking on the destination list button for the source which requires a destination. A list box then appears with options to add or remove nodes from the displayed destination list. For a random list the probability button is grayed out as the probability of sending to each destination in the list is uniformly distributed among the listed nodes.

#### *Connectivity:*

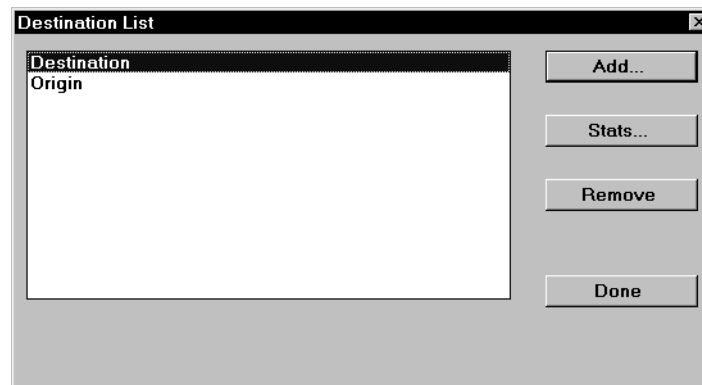
A Random List is allowed as a destination choice for all types of message generators, other than a Response Source and an Answer Message Command. For these latter two types of generators the destination is implicitly the sender of an inbound message.

A destination is any other node in the model, including nodes in the local subnetwork, remote subnetworks or the backbone. Subnetwork node names are prefixed with the subnetwork name. Destination nodes cannot include routers or ATM switches. Messages may be sent to SELF (i.e., the originating node). In this case the node will be busy by the packetizing delay and packet switching times for the packets in the message, but the packet will not be placed in an output port buffer as it is already at its destination.

If a traffic generator should only send to a single destination, this may be entered as a Random List with a single entry.

#### *Editing:*

Edit the details of the destination list from the Destination List dialog box. You have the options to add a new destination to the list, or remove an existing destination from the list. The Set Probability button does not appear on a random list because the probability of sending to each destination is automatically calculated based upon the number of entries in the list.



**Destination List Dialog Box**



**Execution:** When a message, session or call is scheduled at some originating node, a destination has to be selected. If a Random List is in use then a statistical pick is made from the destinations in the list. All destinations have equal weight.

**Reporting:** Node Received Message Count.

**Fields:**

**Destination Node Name** The names of the destination nodes as defined on their dialog boxes. COMNET III makes references to the actual nodes, so that if the node's name changes, its name in the list will be changed automatically.

## 4. Modeling Constructs

### 4.18 Destination: Random Neighbor

<b><i>Purpose:</i></b>	<p>A random neighbor destination list is automatically constructed for the originating node by COMNET III. In this case the nodes that are reachable within one hop from the origin in the same subnet are considered neighbors. When a destination is required one of the neighbors will be picked at random, all neighbors having equal probability.</p> <p>Note: The list of nodes that are one hop away may change as the model is modified and thus the random neighbor may send to unexpected nodes in future modifications of the model. For more control over destinations, use a random list.</p>
<b><i>Creating:</i></b>	<p>A random neighbor list does not have to be explicitly entered. Once you have selected Random Neighbor in the “Type” box, then COMNET III will construct internally the nodes in the destination list.</p>
<b><i>Connectivity:</i></b>	<p>Random neighbors are nodes found over 1 hop in the local network (i.e., backbone or subnetwork). It will not include routers or ATM switches. It will not include SELF.</p> <p>It is a verification error if a Random Neighbor destination is selected, and at least one one-hop destination does not exist.</p>
<b><i>Editing:</i></b>	<p>N/A</p>
<b><i>Execution:</i></b>	<p>When a message, session or call is scheduled at some originating node, a destination has to be selected. If Random Neighbor is in use then a statistical pick is made from those destinations reachable in one hop. All such destinations have equal weight.</p>
<b><i>Reporting:</i></b>	<p>Node received message counts.</p>
<b><i>Fields:</i></b>	<p>N/A</p>

## 4.19 Destination: Round Robin List

<b><i>Purpose:</i></b>	The Round Robin list forms an ordered sequence of destinations. The first time a destination is chosen from the list, the first member will be picked, the second time it will be the second member, and so on. After the last member, the first will be picked again. Ordering of the list is based on the order in which you add them to the list, unless you change it using the Set Order button.
<b><i>Creating:</i></b>	Destination lists are created by clicking on the destination list button for the source which requires a destination. A list box then appears with options to add or remove nodes from the displayed destination list. For a Round Robin List the probability button is grayed out as the probability of sending to each destination in the list is governed by which destination is the next in the list order.
<b><i>Editing:</i></b>	Edit the details of the destination list from the Destination List dialog box. You have the options to add a new destination to the list, or remove an existing destination from the list. The Set Probability button does not appear on a Round Robin List because the probability of sending to each destination is based upon the order of the destination entries in the list.
<b><i>Connectivity:</i></b>	<p>A Round Robin List is allowed as a destination choice for all types of message generators, other than a Response Source and an Answer Message Command. For these latter two types of generators the destination is implicitly the sender of an inbound message.</p> <p>A destination is any other node in the model, including nodes in the local subnetwork, remote subnetworks or the backbone. Subnetwork node names are prefixed with the subnetwork name. Destination nodes cannot include routers or ATM switches. Messages may be sent to SELF (i.e., the originating node). In this case the node will be busy by the packetizing delay and packet switching times for the packets in the message, but the packet will not be placed in an output port buffer as it is already at its destination.</p> <p>If a traffic generator should only send to a single destination, this may be entered as a Round Robin List with a single entry.</p>
<b><i>Execution:</i></b>	<p>When a message, session or call is scheduled at some originating node, a destination has to be selected. If a Round Robin List is in use then the destination that is the first member will be chosen. The next message, session, or call that is scheduled will be sent to the second member in the destination list. Each member of the destination list will be chosen in turn.</p> <p>If the selected destination is a group node, or if the only list member is a group node, the specific destination in the group will be chosen by the same Round Robin algorithm.</p>
<b><i>Reporting:</i></b>	Node Received Message Count.
<b><i>Fields:</i></b>	
<b>Destination Node Name</b>	The names of the destination nodes as defined on their dialog boxes. COMNET III makes references to the actual nodes, so that if the node's name changes, its name in the list will be changed automatically.

## 4. Modeling Constructs

### 4.20 Destination: Weighted List

***Purpose***

A weighted list destination is a list of destination names, one of which is picked at random when a destination is required. All destinations in the list have an assigned probability, so that you can specify which is the most likely destination. The total of all the probabilities must add up to 1.

***Creating:***

Destination lists are created by clicking on the destination list button for the source which requires a destination. A list box then appears with options to add or remove nodes from the displayed destination list. A random list includes a probability button which is used to set the probability of the node being picked as the destination.

***Connectivity:***

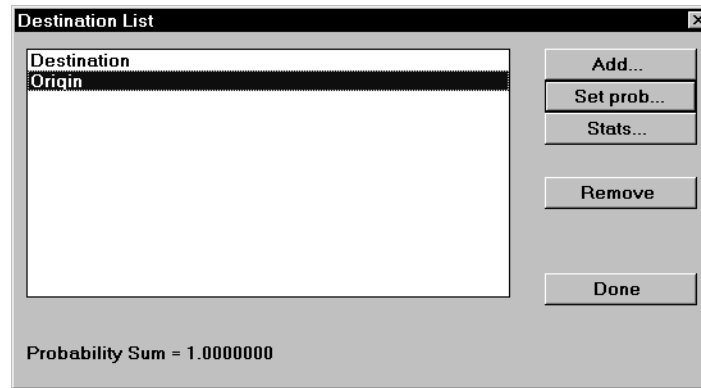
A Weighted List is allowed as a destination choice for all types of traffic generators, other than a Response Source and an Answer Message Command. For these latter two types of generators the destination is implicitly the sender of an inbound message.

A destination is any other node in the model, including nodes in the local subnetwork, remote subnetworks or the backbone. Subnetwork node names are prefixed with the subnetwork name. Destination nodes cannot include routers or ATM switches. Messages may be sent to SELF (i.e., the originating node). In this case the node will be may busy by the packetizing delay and packet switching times for the packets in the message, but the packet will not be placed in an output port buffer as it is already at its destination. For each destination entered in the Weighted List the probability of picking that destination must be entered.

If a traffic generator should only send to a single destination, this may be entered as a Weighted List with a single entry whose probability of selection is 1.

***Editing:***

Edit the details of the list from the appropriate dialog box. Once a destination has been added to the list, the probability of picking it must be set. The default probability is 1, which means that when a new entry is added the sum of all probabilities will be greater than 1 (except on the first entry). This will result in a verify error unless the new entry probability is set to an appropriate value such that the sum total of all probabilities in the list is 1.



Destination List Dialog Box



Set Probability Dialog Box

**Execution:**

When a message, session or call is scheduled at some originating node, a destination has to be selected. If a Weighted List is in use then a statistical pick is made from the destinations in the list using the relative weightings as specified by the user.

**Reporting:**

Node Received Message Counts.

**Fields:****Destination Node Name**

The names of the destination nodes as defined on their dialog boxes.

**Probability**

The probability of the particular node being picked—between 0 and 1. The sum total of all probabilities in the list must equal 1. This is a verification check.

## 4. Modeling Constructs

### 4.21 Distributions: System

#### *Purpose*

COMNET III provides built-in probability distributions, each of which provides a stream of random samples on successive executions. The value of the sample is distributed according to the type and parameters of the distribution.

The basic random number generation algorithm is based on a multiplicative congruence technique which uses a starting seed to mathematically generate a “random” number between 0 and 1 and the next seed. On the next invocation the new seed is used so a different “random” number and new seed are then generated.

The starting seeds are initialized identically between one run and the next so COMNET III will produce identical results if a model is repeated (unless you change some model parameter).

The “random” number sequence is therefore pseudorandom. In other words, even though it is repeatable and mathematically derived it does pass the appropriate statistical tests to qualify as a random number sequence.

On a 32 bit machine, the random number sequence does not repeat (i.e., a seed value is not repeated) inside  $2^{32}-1$  pulls.

To provide multiple, independent random number generators so that different sequences of random numbers can be used for different modelling constructs, up to 99 streams may be used in COMNET III. Each stream is defined as having its own starting seed and so each stream produces its own unique sequence of random numbers.

Not all interarrival times, message sizes etc., are uniformly distributed between 0 and 1. Consequently a number of distribution functions are provided which manipulate the basic random number pulls into a weighted distribution which can then be used to represent model data.

#### *Creating:*

To use a distribution, select “Probability Distribution” when a choice is made available (such as interarrival times, message sizes, retry times, etc.). A drop down list then appears which has some sample distributions with their associated parameters set to some default value. Pick the closest one.

You can then edit the numbers in the text box where the distribution appears directly, or you can click on the adjacent button with two dots and obtain a dialog box with full names and parameter prompts for the selected distribution.

From this latter dialog box you may also elect to graph the distribution with the parameters you have set.

#### *Connectivity:*

N/A

#### *Editing:*

Edit the details of the distribution directly from its text box, or pull up the dialog box with the double-dot button.

#### *Execution:*

N/A

#### *Reporting:*

N/A

***Fields:***

The various distributions with their fields are as follows:

<b>Beta</b>	Shape1, Shape2, Min, Max, Stream
<b>Erlang</b>	Mean, Shape, Stream
<b>Exponential</b>	Mean, Stream
<b>Gamma</b>	Mean, Shape, Stream
<b>Geometric</b>	Min, Mean, Stream
<b>Hyperexponential</b>	Mean1, Mean2, Prob Mean1, Stream
<b>Integer</b>	Min, Max, Stream
<b>Lognormal</b>	Mean, Standard Deviation, Stream
<b>Normal</b>	Mean, Standard Deviation, Stream
<b>Poisson</b>	Mean, Stream
<b>Triangular</b>	Min, Max, Mode, Stream
<b>Uniform</b>	Min, Max, Stream
<b>Weibull</b>	Shape, Scale, Stream

Graphical representations of the distributions are shown in the appendix.

## 4. Modeling Constructs

### 4.22 Distributions: Table

**Purpose**

It may be that the data you want to represent in the model does not follow a mathematically derived statistical distribution—but you have data that you have measured and you want to use statistical picks from it as inputs to the model.

COMNET III provides the ability to define probability tables where your data may be entered and then used alongside the built-in distributions.

**Creating:**

To create a table use the **Define/Tables** menu option. A tabular distributions list dialog box then appears where you can Add, Edit, Copy or Remove probability tables. Pick the Add button. The Table Detail dialog box then appears where you can name the distribution and enter your data values to it.

**Connectivity:**

A table distribution may be used anywhere a system distribution may be used.

**Editing:**

Via the **Define/Tables** menu option. To enter probability or value entries in the table highlight the entry that you want to update by clicking on its box. The value box immediately under the TableType label will then be retitled as “Probability” or “Value” depending on the box you selected and the appropriate value put into the value box. Click with the mouse in this value box and enter the updated value you require. Hit the “Enter” key when finished and the new value will then be put into the appropriate table box.

**Execution:**

N/A

**Reporting:**

N/A

Cum Prob	Value
0.500000	10.000000
1.000000	20.000000

**Table Distribution Dialog Box**  
(50/50 split of values at 10 or 20 only)

**Fields:**

**Name**

An alphanumeric field to identify the table. The name must be unique among tables. The name is later used to reference this distribution when you use it for an interarrival time, message size, holding time, etc. The table name may not con-



tain spaces.

**Table Type**

There are two ways of interpreting the data values that you enter:

**Discrete**

The values that you enter are taken as observation points, together with their cumulative probabilities. When COMNET III picks from the table any one of the entered values may be picked with the probability determining its likelihood.

**Continuous**

The values that you enter are taken as observations on the curve which is the envelope of the distribution. When COMNET III picks from the table it will interpolate between the adjacent points and pick any value that is on the curve using the interpolated probability.

**Probability**

The probability of the value occurring. The probabilities are entered cumulatively—i.e., the last probability must equal 1. As a consequence, each probability must be equal or greater than the previous one in the list.

**Value**

The associated value to the probability. Each value must be greater than the one preceding it in the list.

## 4. Modeling Constructs

### 4.23 Distributions: User

#### *Purpose*

It may be that the data you want to represent in the model follows one of the COMNET III distributions with particular parameter values. Rather than re-entering this in several places you can make a User Distribution, which is a named version of one of the standard distributions with its parameters set.

This named distribution then appears in the drop down list of probability distributions and may be picked equally with any of the other choices.

One benefit of this approach is that the User Distribution is only defined in one place. Therefore, if you change the parameters then all places where the distribution is used will see the change. Also, all user distributions are accessible from single user-distributions list.

For instance, you can make a user distribution called Generator which uses an Exponential(10,2) standard distribution. You may then use Generator on 10 message generators to control the interarrival time. Then, if you want to double the traffic on these generators, you only need to edit Generator and change it once to Exponential(5,2) and this will then be applied to all the places where Generator is used.

#### *Creating:*

To create a user distribution use the Define/User Distribution menu option. A user distributions list dialog box then appears where you can Add, Edit, Copy or Remove user distributions. Pick the Add button. You then enter the name of the user distribution and pick from one of the standard COMNET III distributions and enter its parameters.

#### *Connectivity:*

A Named Distribution may be used anywhere a system distribution may be used.

#### *Editing:*

Via the Define/User Distribution menu option.

#### *Execution:*

N/A

#### *Reporting:*

N/A

#### **Fields:**

##### **Name**

An alphanumeric field to identify the user distribution. The name must be unique between user distributions. The name is later used to reference this distribution when you use it for an interarrival time, message size, holding time, etc.

##### **Probability Distribution**

One of the standard COMNET III distributions with its parameters set.  
(See *Distributions: System*)

## 4.24 External Model Files

*Purpose:*

COMNET III 1.1 supports an easy-to-read ASCII model file format that allows creation of model files without using the COMNET III graphical user interface. These files are loaded into COMNET III using a new Import command on the File menu. In addition, there is a Merge command that merges the objects described in an external model file with a model that has already been loaded in COMNET III. The external file format is described in a separate document, *COMNET III External Model File Format*, which is available upon request.

The external model file is used by CACI to provide the interface to Network Management Systems. After extracting the network topology from Cabletron Spectrum, HP OpenView, or IBM NetView 6000, the extracted file is converted by a COMNET III utility into an external model file, which is then imported directly into COMNET III. The conversion of the topology file to the external model file is performed automatically when the Import command is used for a topology file from one of the above named Network Management Systems.

The external model file is also used to provide an interface between the Network Analysis Center, Inc.'s WinMIND network design and pricing software and COMNET III. After using COMNET III to design and model your network, you can import data from COMNET III into WinMIND using COMNET III's external model file format, to produce a least cost network design.

## 4. Modeling Constructs

### 4.25 External Traffic Files

*Purpose:*

A model can have external message, session, and call sources that are triggered by an external event file. Each model can have an external message file, an external session file, and an external call file. An external event file contains a sequence of records in chronological order. Each record has a time stamp, an origin, a destination, an ID, and a “size.” The “size” represents the message size for the message file, the number of messages for the session file, and the call holding time for the call file.

The model has a list of external message source prototypes (and a list of external session and call sources). COMNET III uses these prototypes as templates to automatically add message sources where they are needed as indicated by the originating nodes in the external message file. An event record is mapped to a particular type of external message source at an originating node by the ID field on the record.

The mapping of different types of messages to different types of external message sources lets the user specify additional properties (such as priority, transport layer, or routed protocol) for external traffic that may vary by the type of message. The traffic in the external file can be automatically scaled.

COMNET III provides an interface to a utility called AutoBaseliner that is used to specify the format of the user’s external file. Built-in format specifications are provided for trace files from Network General Sniffer and Expert Sniffer, Hewlett Packard’s NetMetrix, Wandel & Goltermans Domino Analyzer, Frontier Technology’s NetScout RMON Probe, and CastleRock SNMP RMON probe. AutoBaseliner is also used to match names in each external file with node names used in the COMNET III model. After specifying the file format and matching names, AutoBaseliner produces an event file “linked” to model names; the “linked” file is what actually drives the simulation.

## 4.26 Flow Control

### *Purpose:*

A number of Flow Control algorithms are built into COMNET III to allow for end-to-end flow control modeling between origin and destination. These include fixed window, sliding window, and SNA pacing control algorithms. When a packet creation attempt is made at the origin, a check is made that any outstanding acknowledgments have been received. If an ACK is outstanding then the packet creation attempt is passed over until the ACK arrives. This aids in congestion control in the network, and models the error checking mechanisms found in some protocols (e.g., X.25).

Link level flow control is not modeled in COMNET III.

### *Creating:*

Flow control algorithms are accessed via the Transport Protocol screens where the appropriate flow control setting may be made.

Setting the flow control to “None” turns off any flow control processing.

### *Connectivity:*

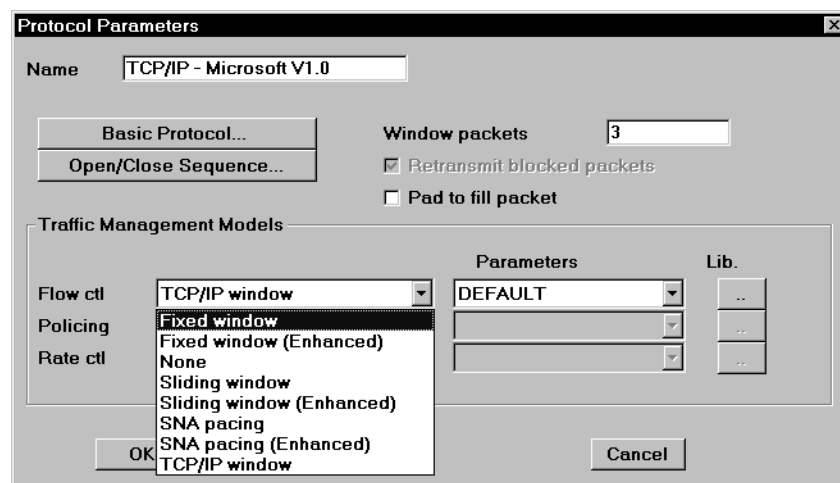
Flow control may be used for packet-switched message traffic arising from Message Sources, Session Sources, Response Sources, Transport Commands, Session Setup Commands or Answer Commands.

For flow control to be effective, retransmission of blocked packets must be allowed (otherwise window counters would become “corrupt” if packets in transit were lost).

Also the message must have a single destination. In other words, flow control is not supported for multicast messages.

### *Editing:*

Via the appropriate Transport Protocol screen.



**Flow Control Dialog Box  
(Part of Transport Protocol Settings)**

### *Execution:*

See Fixed Window, Sliding Window, or SNA Pacing sections.

## 4. Modeling Constructs

**Reporting:**

N/A

**Fields:**

(Common to all flow control methods)

**Window Size**

Acknowledgments are generated at the destination depending upon the flow control algorithm and the window size setting. Once the originating node has sent a window size worth of packets, it will wait for the acknowledgment to arrive before it may generate the next packet in the message.

If the originating node attempts to generate a packet and the ACK has not been received, it will pass over the generation attempt and return to attempt the generation after other pending tasks have been finished (see Node Scheduling).

If there are no other tasks to complete then the node will wait for the ACK packet to be received and then generate the next packet in the message.

Some protocols use window sizes specified in bytes. In those cases, use equivalent (or approximate) number of full-sized packets that fit in the window. Other protocols (such as IP) advertise windows in bytes but the internal logic adjusts the size in full packets for efficiency reasons—thus these windows are measured accurately as a number of packets.

**End-To-End ACK Bytes**

The size of the ACK packet generated at the destination. This will be routed back to the origin based upon the effective network routing protocols, and the routing class and transport protocol settings of the inbound packet.

**Retransmit time (ms)**

The time before retransmission of a blocked packet is attempted.

**ACK Priority**

The priority of the ACK packets. This is used for ordering packets in input and output buffers at nodes en route back to the origin. ACKs are usually higher priority than user data packets.

**Retransmit Blocked Packets**

This flag must be on for flow control to be effective. If a data or ACK packet blocks en route because of link/node failure, or because of buffer congestion, the packet transmission will be reattempted from the origin after the retry interval.

**Fields for Enhanced Flow Control:**

**Error Control Options**

There are three error control options available for the enhanced flow-control windows. The selective repeat will retransmit just the packet that is blocked (similar to how the simple windows handle blocked packets). The go-back-n option will retransmit the entire window of unacknowledged packets, which for fixed and SNA-pacing windows is the entire window, but for sliding windows it is just that portion of the window that is still outstanding. The fast-recovery option applies just to sliding windows (such as TCP/IP) that would retransmit the entire window unless it detects multiple duplicate acknowledgements that indicates only one packet needs to be retransmitted.

**Holding ACKs for More Data**

Some sliding window protocols have a provision to delay the acknowledgement in order to improve the chance to piggy-back data on the acknowledgement. COMNET III does not model the piggy-back of reverse data on the acknowledgement but it does model this delay because it has an impact on overall message delay. The Hold-1 option allows for the delay to be interrupted if there is a sec-

ond acknowledgement waiting, while the Hold-all option accumulates all acknowledgements until the delay has expired. When an acknowledgement is delayed, the transmitted acknowledgement will acknowledge all packets that have been received during that delay.

**Retransmit Timeout Calculated from Estimated Round-Trip Time**

Because the sliding-window algorithms acknowledge each packet (or at least most packets), they have an opportunity to measure round-trip packet delay and use that delay to adjust the retransmission time-out timer. TCP/IP in particular does this. There are two options for estimating the round-trip time and using that estimate for the retransmission time-out. The first option estimates the round-trip time and uses a multiple of that time for the retransmission time. The second option estimates both the average round trip time and its standard deviation and then uses the average plus a multiple of the standard deviation for the retransmission time.

**Retransmit Timeout Backoff** This option applies if a packet is blocked when it is retransmitted. When active, this option will double the previous retransmission time to determine the retransmission time for successive retransmissions of the same packet.

## 4. Modeling Constructs

### 4.27 Flow Control: Fixed Window

<b><i>Purpose:</i></b>	To model fixed window flow control.
<b><i>Creating:</i></b>	Via the Transport Protocol screens.
<b><i>Connectivity:</i></b>	For packet oriented message traffic.
<b><i>Editing:</i></b>	Via the Transport Protocol screens.
<b><i>Execution:</i></b>	The origin creates and transmits a packet. This is repeated until the number of packets sent equals the window size. The last packet to reach the destination is acknowledged. When the ACK is received back at the origin another window size worth of packets may be created and transmitted.
<b><i>Reporting:</i></b>	N/A
<b><i>Fields:</i></b>	<i>See Flow Control and Transport Protocol</i>



## 4.28 Flow Control: Sliding Window

<b><i>Purpose:</i></b>	To provide an appropriate model of Sliding Window Flow Control that runs fast.
<b><i>Creating:</i></b>	Via the Transport Protocol screens.
<b><i>Connectivity:</i></b>	For packet oriented message traffic.
<b><i>Editing:</i></b>	Via the Transport Protocol screens.
<b><i>Execution:</i></b>	The Sliding Window Flow Control allows up to the window size (n) of packets to be transmitted without an acknowledgement. Each acknowledgement allows the window to advance by one, so that the window slides by ack sequence number instead of jumping the full window size as Fixed Window Flow Control does.
<b><i>Reporting:</i></b>	N/A
<b><i>Fields:</i></b>	<i>See Flow Control and Transport Protocol</i>

## 4. Modeling Constructs

### 4.29 Flow Control: Jumping Window

<b><i>Purpose:</i></b>	The jumping window measures the burst based on accumulating a burst size with each packet, but then clearing that accumulation at fixed intervals. In contrast to the sliding window which always accounts for all traffic in a previous period of time, the jumping window clears its burst periodically and thus periodically loses memory of the recently accepted packets. As a result, the jumping window is the most lenient of the three burst measurement algorithms.
<b><i>Creating:</i></b>	Via the Transport Protocol screens.
<b><i>Connectivity:</i></b>	For packet oriented message traffic.
<b><i>Editing:</i></b>	Via the Transport Protocol screens.
<b><i>Execution:</i></b>	The origin creates and transmits a packet. This is repeated until the number of packets sent equals the window size. Every packet to reach the destination is acknowledged. When an ACK is received back at the origin another packet may be created and transmitted.
<b><i>Reporting:</i></b>	N/A
<b><i>Fields:</i></b>	<i>See Flow Control and Transport Protocol</i>

### 4.30 Flow Control: Leaky Bucket

<b><i>Purpose:</i></b>	The leaky bucket algorithm (or the Generalized Cell Rate Algorithm, GCRA, for ATM) is a common algorithm for measuring bursts because it is simple to implement and it is not as lenient as the jumping window in terms of allowing excess traffic. It is similar to the jumping window algorithm except that it periodically subtracts off an amount equal to the rate times the burst-interval. This maintains some memory of the previous burst interval, especially if that burst exceeded the value of the burst rate times the burst interval. It is more strict than the jumping window in terms of accepting traffic but not as strict as the sliding window.
<b><i>Creating:</i></b>	Via the Transport Protocol screens.
<b><i>Connectivity:</i></b>	For packet oriented message traffic.
<b><i>Editing:</i></b>	Via the Transport Protocol screens.
<b><i>Execution:</i></b>	The origin creates and transmits a packet. This is repeated until the number of packets sent equals the window size. Every packet to reach the destination is acknowledged. When an ACK is received back at the origin another packet may be created and transmitted.
<b><i>Reporting:</i></b>	N/A
<b><i>Fields:</i></b>	<i>See Flow Control and Transport Protocol</i>

## 4. Modeling Constructs

### 4.31 Flow Control: SNA Pacing

<b><i>Purpose:</i></b>	To model SNA Pacing flow control.
<b><i>Creating:</i></b>	Via the Transport Protocol screens.
<b><i>Connectivity:</i></b>	For packet oriented message traffic.
<b><i>Editing:</i></b>	Via the Transport Protocol screens.
<b><i>Execution:</i></b>	With SNA Pacing flow control, a pacing counter is initialized to the window size. The pacing counter is decremented every time a packet is created. Packet creation stops when the counter reaches zero. Only the first packet in a window is acknowledged. When the ACK packet is received back at the origin the pacing counter is incremented by the window size and another window of packets may be created and transmitted.
<b><i>Reporting:</i></b>	N/A
<b><i>Fields:</i></b>	<i>See Flow Control and Transport Protocol</i>

## 4.32 Flow Control: TCP/IP Window

<b><i>Purpose:</i></b>	The TCP/IP window is based upon the sliding window plus adds the specific congestion window algorithm to avoid congesting a network. It provides a realistic flow control option for TCP/IP that explicitly models the adaptive window size of the TCP/IP protocol.
<b><i>Creating:</i></b>	Via the Transport Protocol screens.
<b><i>Connectivity:</i></b>	For packet oriented message traffic.
<b><i>Editing:</i></b>	Via the Transport Protocol screens.
<b><i>Execution:</i></b>	A TCP/IP connection starts with a window size of one packet and then increases the window by one for each acknowledgement to result in an exponentially increasing window. If congestion has been previously detected on this connection, the window will increase until it reaches one half of the window size when the congestion was detected, otherwise it will increase to the full available window for that connection. For the congestion case, the window advances approximately one packet per window of packets so that the window grows more slowly to avoid any congestion that may still be in the network. If congestion is detected, the window will return to one unless duplicate acknowledgements indicate that only one packet has been lost – a condition known as fast recovery.
<b><i>Reporting:</i></b>	N/A
<b><i>Fields:</i></b>	<i>See Flow Control and Transport Protocol</i>

## 4. Modeling Constructs

### 4.33 IEEE 802

The IEEE 802 standards are listed here for your reference.

- 802.1 Higher Layer Interface
- 802.2 Logical Link Control (additional overhead on frame)
- 802.3 CSMA/CD Networks (CSMA/CD link)
- 802.4 Token Bus Networks (Token-passing)
- 802.5 Token Ring Networks (Token-passing)
- 802.6 Metropolitan Area Networks
- 802.7 Broadband Technical Advisory Group
- 802.8 Fiber Optic Technical Advisory Group
- 802.9 Integrated Voice And Data LAN Interface
- 802.10 Standard For Interoperable LAN Security
- 802.11 CSMA/CA Wireless LAN

## 4.34 Link

**Purpose:**

A link to which nodes may be connected by arcs. Types of links include Aloha, CSMA, CSMA/CD, Polling, Point-to-Point, and Token Passing.

**Link Detail Dialog Box**

**Creating**

To create a link, choose one from the palette and drag it onto the background. You can also create one by choosing Create/Links... from the menu bar.

**Connectivity:**

For details about which types of links connect to which types of nodes, see the section on the appropriate node.

**Editing:**

To edit a link, either click once on the link to select it and then choose Edit/Detail, or double-click on it. Both methods bring up the Link Detail dialog box.

**Fields:**

**Name:**

The name of the link. The name must be unique in the backbone or subnetwork.

**Icon**

The name of the icon used to represent the link.

**Type**

The type of link: Aloha, CSMA, CSMA/CD, Point-to-Point, Polling, Token Passing, or one defined by the user.

**Parameters**

Parameters for that type of link. Parameters may include specifications for the number of circuits, bandwidth/circuit, session limit, frame size, retry parameters, etc., and will depend on the link type.

**Control Node**

The name of the node which is the control node for the link. A control node has an independent, contention free channel to communicate with the other nodes on the link. The control node is deselected unless the control channel check box is ticked on the link parameters screen.

**Time to failure (min)**

Links may be made to fail at a time determined by a probability distribution.

**Time to repair (min)**

The time it takes a link to be repaired can be set by a probability distribution.

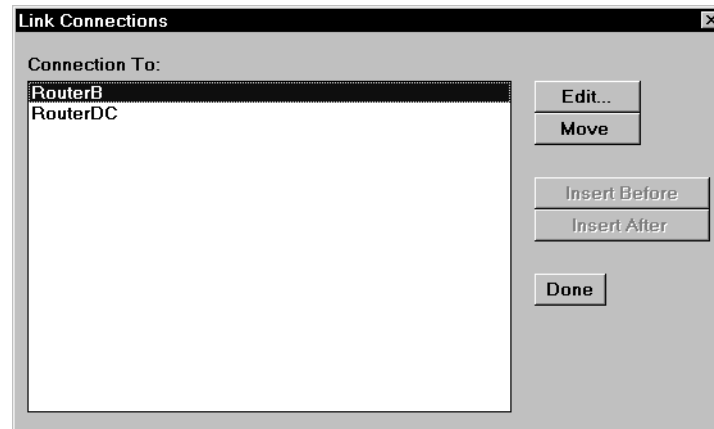
#### 4. Modeling Constructs

##### Time of next state change

A time at which the state of the link will be toggled: if the link is currently up the link will be brought down; if the link is currently down it will be brought back on-line. The time is given in terms of the simulation time. For example, if the Time of the next state change is 30 seconds, then 30 seconds into each replication the state will be changed.

##### Current state

Whether the link is currently operating (up) or failed (down). This can be toggled during the simulation by double-clicking on the link to bring up the link detail dialog box and editing this field.



##### Connections...

This button allows you to specify the ordering of nodes connected to a specific link. The order determines the node sequence for token-passing and polling. For point-to-point links with different transmission speeds in each direction, the order determines which speed applies to each transmitting node.



## 4.35 Link: Binary Exponential Backoff Parameters

**Purpose:**

The binary exponential backoff algorithm is defined by the IEEE for computing a retry time for nodes connected to collision detecting links - CSMA and CSMA/CD Links. Upon collision each node has to calculate a retry time so that it may attempt transmission again. Each node should pick a different time to avoid infinite collisions. The time is an integral number of time slots calculated as explained in the Execution section below. This distribution is available only on collision links: Aloha, CSMA, CSMA/CD.

**Creating:**

The binary exponential backoff algorithm is built into COMNET III. It may be picked as one of the options when a retry time is required.

**Connectivity:**

N/A

**Editing:**

Via the appropriate dialog box (e.g., ALOHA link, CSMA link etc.)

**IEEE Binary Exponential Backoff Dialog Box  
(as part of CSMA/CD parameters)**

**Execution:**

The binary exponential backoff algorithm doubles the retry interval for a frame every time the frame collides. Once the frame has collided 10 times, the maximum possible retry time stops increasing. Assuming the number of collisions is below some maximum value (the retry limit), the retry time is computed as a constant offset time plus the slot time multiplied by a random integer between 0 and  $2^{\min[10, \text{Collision Count}]}$ . If a frame has exceeded the retry limit, the retry time is given by the limit delay.

**Reporting:**

N/A

**Fields:**

## 4. Modeling Constructs

### Slot Time

The slot time is chosen to be large enough to guarantee that two nodes starting to transmit at different slot time boundaries will not collide again.

This results in an interval at least equal to the round trip propagation time on the link plus the jam time.

IEEE 802.3 sets the slot time at 0.0512 milliseconds to accommodate the longest path allowed by 802.3 (2.5km and 4 repeaters).

The retry time computed by the Binary Exponential Backoff algorithm will result in an integral number of slot times, or the limit delay if the retry limit has been reached.

### Offset

A constant added to the IEEE 802.3 retry time.

### Retry Limit

After a frame collides this number of times, the node stops trying and waits the limit delay.

### Limit Delay

When the number of collisions for a frame reaches the retry limit, the transmitting node waits the limit delay before trying again. The collision count is set back to 0.

### Stream

The random number stream to pick the random integers used to multiply the slot time in the retry time calculation.

## 4.36 Link: Framing Characteristics

### *Purpose:*

When packet transmission is modeled across a link, packets are broken down into transmission frames which are then transmitted across the link. When all frames from the packet have been received at the other end of the link then the packet is reassembled and placed in the input buffer of the receiving node.

The framing characteristics are the place where the layer 2 characteristics of the link are specified. While COMNET III does not include a detailed representation of all of the layer 2 protocol on the link (e.g., time outs, CRC checks, etc.) the framing characteristics contain sufficient details of the layer 2 protocol characteristics such that the transmission time for packets and overall link loading may be predicted.

### *Creating:*

Framing characteristics are defined on the link parameters screens.

### *Connectivity:*

All types of links may have framing characteristics set.

### *Editing:*

Via the link parameters screens.

	Packets, Frames, Cells	Circuit-Switched Calls
Parameter set name	DEFAULT	
Number of circuits	1	1
Bandwidth/circuit (kbps) (from node X)	1536.000	1536.000
Bandwidth/circuit (kbps) (from node Y)	1536.000	
Bandwidth reserved for 1-hop calls (%)		0.0000000
Propagation (ms)	0.000	
Session limit	1024	
Frame min (bytes)	0	
Frame max (bytes)	0	
Frame overhead (bytes)	0	
<input type="checkbox"/> Frame assembly		
Frame error prob	0.00000000	
<input checked="" type="checkbox"/> Automatically retransmit frame errors		

Link Parameters Dialog Box For A Point-to-Point Link

### *Execution:*

The packets are queued in priority and FIFO order in the output buffer of the port connecting the node to the link. Depending on the MAC protocol (i.e., token passing, CSMA, Point-to-Point, etc.), the node will at some point in time gain access to the link such that the first packet may be transmitted across the link.

Based on the framing characteristics, the packet is broken into a number of transmission frames. Each frame has a size. The link has a transmission rate. The time delay for transmission is then calculated by dividing the frame size by the transmission speed. The link is made busy for this transmission time and then the link

## 4. Modeling Constructs

is made idle. The frame is then delayed by the propagation delay. After this time the frame is considered to have arrived at the next node, but it is not yet placed in the input port buffer.

This calculation is repeated for all the frames (Note: there may be interframe gaps and contention for the link for each frame if the link has a multiple access protocol such as CSMA). When the last frame arrives, the packet is reassembled and placed into the receive buffer on the input port at the receiving node, and the packet releases its buffer space at the sending node.

Frame Delay is the time for each frame to go from one node to the next: it includes transmission, propagation, and LAN contention delays.

Link utilization is the fraction of the time that the link is busy transmitting a frame.

### ***Reporting:***

Frame transmission is shown on the following reports:  
Link Delays and Utilization  
Random Access Link Performance

### ***Fields:***

#### **Frame Min**

Packets smaller than the minimum frame size will be padded up to the minimum frame size from the point of view of calculating their transmission time.

If a large packet is broken down into several full size frames plus an overflow frame, the overflow frame will be padded to the minimum size.

#### **Frame Max**

Large packets are broken down into several full size frames plus an overflow frame. A frame max of 0 bytes is taken to mean that there is no upper limit on frame size. In this case packets will be transmitted in their entirety without being broken into frames.

#### **Frame Overhead**

The number of bytes used by the link level protocol for error checking, addressing, etc. The detailed link protocol is not modeled other than by the time calculations described in this framing section. Particularly, issues such as timeouts, link level acknowledgments and error recovery are not part of the COMNET III model. The frame overhead is inclusive in the frame min and frame max values.

#### **Frame Error Probability**

The probability that a maximum size frame arrives in error and requires retransmission. If a statistical pull decides that the frame requires retransmission then the link will be made busy for the frame transmission time again.

This parameter is aimed at modelling lines with infrequent errors where retransmission may impose increased workload or increased frame delay on the link. The error detection and recovery mechanisms is selective repeat which is appropriate for low error rate links.

#### **Automatically Retransmit Frame Errors**

Link frame errors can optionally not be corrected at the datalink layer. If a frame error occurs, the packet carried by the frame is treated as blocked, causing the protocol to either retransmit the packet or to simply drop the packet and fail to reassemble the message at the destination.

## 4.37 Link: ISDN and SONET Parameter Sets

*Purpose:*

The parameter set library for point-to-point links has been expanded to include a range of ISDN and SONET parameter values:

**ISDN:**

- ISDN BRI(2B)
- ISDN BRI(2B w/bonding)
- ISDN BRI(2B+D w/bonding)
- ISDN PRI(24B)
- ISDN PRI(H11)
- ISDN PRI(H12)

**SONET:**

- STS-1/OC-1
- STS-3/OC-3/STM-1
- STS-9/OC-9/STM-3
- STS-12/OC-12/STM-4
- STS-18/OC-18/STM-6
- STS-24/OC-24/STM-8
- STS-36/OC-36/STM-12
- STS-48/OC-48/STM-16

## 4. Modeling Constructs

### 4.38 Link: Loading

#### **Overview:**

Simulation models of large networks often divide the traffic into two types: foreground traffic representing detailed models of applications and their protocols, and background traffic representing the existing utilization that competes with the foreground traffic. Such models require a mechanism for modeling background or baseline loading of the network. Often, this load is known only from a measurement of utilization on the link without any information as to the nature of that traffic.

The most accurate model of background traffic is with detailed modeling of the applications that are generating the traffic and the protocols used to generate the packets. This kind of model uses traffic sources (or application sources) to generate the messages and model the associated protocols. This approach provides the flexibility to model the following to a high level of accuracy:

- The message interarrival and the bursts of packets created by protocols.
- The node utilization as well as the link utilization as these are not independent.
- The mix of packet sizes and priorities that will compete for access to the link.
- The destination of the traffic to accurately model the traffic taking different routes due to congestion or changes in link or node states.

Modeling background traffic with message sources is often impractical because the size of the network requires too many message sources to be configured, and the message sources themselves require entry of many detailed attributes. More importantly, it is very common that most of the above details are not known about the background traffic, or that the only aspect known about the traffic is the utilization that is present on the individual links in the network.

Release 1.3 of COMNET III introduces an alternative method for modeling the load on a link by simply identifying the utilization for that link. There are a variety of load models available to allow a choice of compromises between simulation speed and fidelity. Lower fidelity models allow for the practical simulation of larger networks since the computational demands are reduced.

- The highest fidelity load model consists of a generator of random frames that are introduced to the buffers of the link according to the desired level of utilization. The random frame model avoids the modeling of node activity for the traffic, and as a result it can simulate background utilization of the link more quickly than models using message sources for background traffic. This increase in simulation performance comes at a loss of flexibility for modeling the background frames and treats each link load independently and each link load as independent of the load on the node.

- A lower fidelity model waits until an end-to-end packet arrives at the buffer and then computes the utilization of the link from the last time the link was used. This back-fill may result in excess background frames that would not have had a chance to transmit across the link, and these frames would then be transmitted before the foreground packet is allowed to transmit.

- A very low fidelity model uses an analytical algorithm to determine the time required to access the link based on queuing theory equations. This model runs very fast because there are no additional frames to model, but it can be the least accurate, especially if the foreground traffic significantly contributes to the link utilization compared with the background utilization.

The link loading model may use a constant utilization parameter for each link, or it use the call bandwidth as the utilization parameter. In this context, the call source is an abstract model of packet traffic bandwidth requirements between pairs of nodes. Using calls to indicate bandwidth provides the opportunity to model end-to-end background traffic, and to model background load that changes during the course of the simulation. The ability to model end-to-end background traffic is useful for modeling the interdependence of traffic (links will have similar loads at the same time) and for modeling the rerouting of traffic due to failures or repairs of links or nodes.

There is an additional load model for modeling call bandwidth competing with the packet bandwidth on point-to-point links. In earlier releases of COMNET III the point-to-point links could be used in packet networks to model data communications or in circuit-switched networks to model voice or dedicated channel communications, but if they were used in models with both, the resources for the calls and packets were treated as distinct and independent. In particular, this meant that the bandwidth available for transmitting packets was unaffected by the amount of call bandwidth allocated on the link.

In release 1.3, there is a call-based loading model that allows the calls (representing, for example, voice calls) to take bandwidth away from the available bandwidth for packets. This allows the link to be more heavily utilized as more calls use the link, and thus increase the delays for transmitting the frames. Unlike the other load models, the call-based load model does not model random frames utilizing the link, instead it models the call utilization on the link directly and then adjusts the transmission time of individual frames based on the available bandwidth.

#### ***Loaded Link Model:***

In Release 1.3, the loaded link model is available for point-to-point links and the call-loading is available on a demand-assigned multiple-access (DAMA) link. The loading is specified on the initial dialog for the link as shown in figure 1. Automatically loaded links will be supported for the multiple access link types in a future release.

## 4. Modeling Constructs

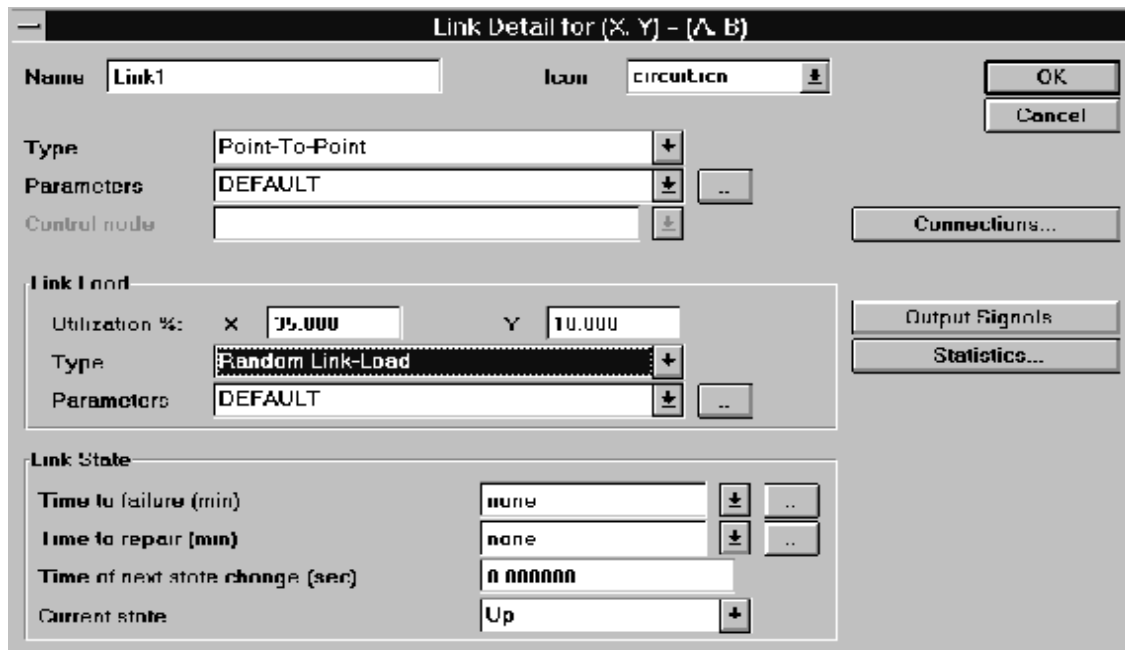


Figure 1. Link properties dialog.

The link load is a property of the link that is distinct from the type. This avoids having to make different type parameters for different kinds of loads. The link load consists of three fields in this dialog: the utilization parameters, the type of load, and the parameters for that type. The utilization parameters are available on the link properties dialog to allow distinct utilization values for each link without requiring maintaining a large number of parameter sets. In general, it is likely that each link in a model will have unique utilization values.

There are five options for modeling the link load:

- 1) "None" is the default and models no load on the link.
- 2) "Random Link-Load" continuously generates random frames on the link.
- 3) "Backfill Random Link-Load" fills in utilization before a packet uses the link.
- 4) "Analytical Link-Load" just calculates the delay to access the link.
- 5) "Call-Based Link-Load" lets calls on the link utilize the packet bandwidth.

For the random frame load models, there is an option in the parameter set to compute the inter-arrival times of the random frames based on the utilization parameter on the link. If the option is on, the dialog will have active one or two utilization fields for specifying the utilization for each transmission line separately. Note that point-to-point links are full-duplex – there is a transmission channel in both directions. Alternatively, the frame inter-arrival times can be specified explicitly, or it can be based on the call bandwidth utilizing that link; in either case, the utilization field of the "Link Properties" dialog will be deactivated.



## Link Load Types:

### Random Link-Load:

The “Random Link-Load” model generates frames randomly and continuously for the duration of the simulation. The random frames may have a specific priority or discard-eligibility for proper modeling of the output buffer to this loaded link. The random frames can have random frame sizes chosen from any distribution or variable. The inter-arrival times of the frames may be exponentially distributed with a mean computed to give the utilization specified on the “Link Properties” dialog. Alternatively, the inter-arrival times may be set explicitly or computed from the call utilization on this link. Figure 2 shows the properties available to customize the behavior of this link load model.

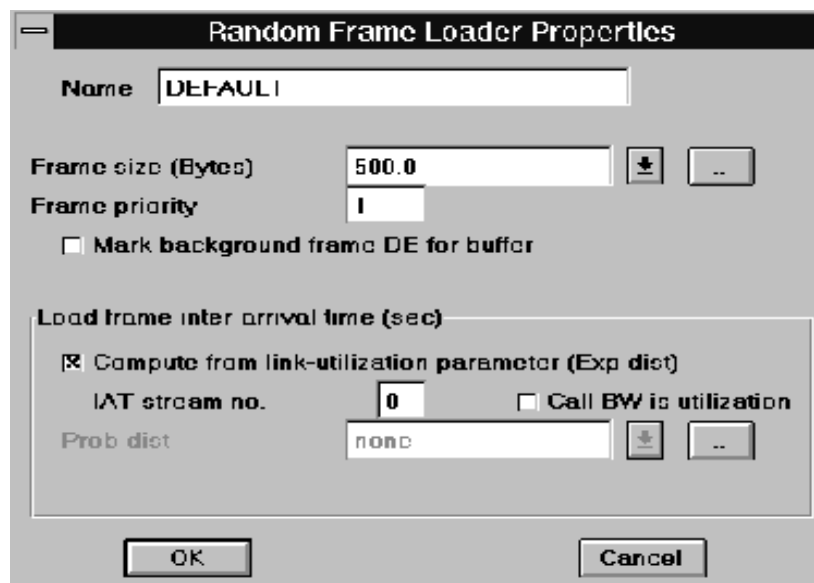


Figure 2. Random Frame Loader Properties

### Backfill Random Link-Load:

The “Backfill Random Link-Load” is similar to the “Random Link-Load” and it uses the same parameter set. The algorithm is different in that the link load is not modeled when there is no traffic trying to use the link. When a packet arrives at the output buffer to this link, this model fills in the background traffic that would have occurred since the last transmission, and it will schedule left-over background frames to be transmitted before the packet gets access to the link. Because this algorithm avoids being scheduled by the simulation clock, it runs substantially faster than the random generator. However, the algorithm becomes less beneficial when the total load increases to nearly full utilization of the link. The “Random Link-Load” model is recommended for modeling highly utilized links.

This load type only works for single circuit links.

### Analytical Link-Load

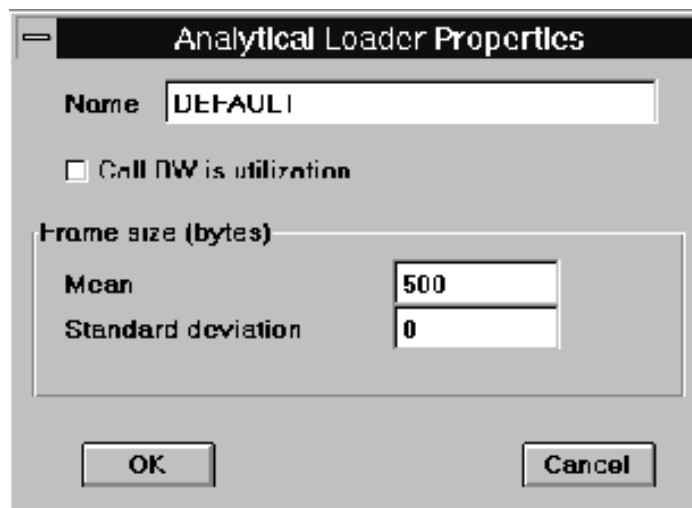
The analytical link-load is a fast algorithm that adds no additional events when a packet tries to access the link. Instead, an access delay based on the M/G/1 queue theory model is computed and the packet gets access to the link after this delay. The M/G/1 model assumes exponential inter-arrival times with a general

## 4. Modeling Constructs

frame distribution and one circuit, thus this model is only available for single packet circuit links. Because no additional events are modeled, this algorithm simulates nearly as quickly as an idle link, even though the packets will be delayed and the link will be utilized appropriate to the utilization modeled.

The applicability or validity of this model is limited to those cases where the foreground traffic introduces low to modest increased utilization on the link. The model will lose accuracy if the foreground traffic utilizes the link much more than the background utilization. The previously described link-load models are better at modeling delays when there is substantial increase in utilization due to the foreground traffic.

The parameters for the “Analytic Link-Load” are shown in Figure 3. Like the random link-load, the utilization may be set in the “Link Properties” dialog or it may be based on the call utilization on the link. The general distribution model requires the specification of the frame size mean and standard deviation to compute the delay for accessing the link.



The image shows a dialog box titled "Analytical Loader Properties". It has a "Name" field containing "DEFAULT". Below this is a checkbox labeled "Call BW is utilization" which is not checked. Underneath is a section titled "Frame size (bytes)" which contains two input fields: "Mean" with the value "500" and "Standard deviation" with the value "0". At the bottom of the dialog are "OK" and "Cancel" buttons.

Figure 3. Analytical Link-Load Properties

### Call-based Link-Load

The “Call-based Link-Load” is used to model a situation where the link bandwidth is shared between the call bandwidth and the packet transmission. The bandwidths of the packet and call circuits are specified separately in the type parameters for the link.

The “Call-based Link-Load” has specific parameters for modeling how the call bandwidth shares the packet bandwidth. These parameters are shown in Figure 4. These parameters are described separately in the following paragraphs.

Figure 4. Call-based Link-Load Properties

The call overhead parameter allows for modeling cases where each call requires additional bandwidth on the physical link to support the call on a shared channel.

The checkbox for “Transmit on partial call circuit” provides an option for packets to use a call circuit if some of the bandwidth on this channel is already allocated to calls. Some systems require a channel to be reserved for calls if only a part of the channel is currently carrying a call.

The checkbox for “Transmit on partial packet circuit” provides an option for packets to use a packet channel that is partially carrying call bandwidth. If there are multiple packet channels available, idle packet channels are used first and the partially filled channel will only be used as a last resort and then only if the checkbox allows it.

For example, suppose a T1 link aggregates calls into 64 kbps circuits and each packet transmission uses the combined remaining bandwidth after subtracting 64 kbps for each circuit that is carrying calls. To model this kind of link operation, set the call and packet bandwidth parameters as follows:

Call bandwidth: 24 circuits of 64 kbps

Packet bandwidth: 1 circuit of 1536 kbps

Set the call-based link loading options as follows:

Transmit on partial call circuit = FALSE

Transmit on partial packet circuit = TRUE

In this example, “Transmit on partial call circuit = FALSE” would mean that a packet could not use the available bandwidth on a 64 kbps circuit if the circuit were already carrying a 32 kbps call. “Transmit on partial packet circuit = TRUE” would mean that a packet could use 768 kbps of bandwidth if 12 circuits were in use for calls.

#### 4. Modeling Constructs

Transmitting a packet on a channel that is partially occupied by calls requires an algorithm for the delay that the packet experiences on the link. There are two algorithms available.

1) One algorithm computes a transmission time for the packet based on the channel bandwidth available at the start of the packet transmission. This algorithm simulates efficiently but it can result in errors when the packet can take advantage of bandwidth opened up (or be penalized if more bandwidth is taken away) by calls during the transmission of the packet.

2) A second algorithm slices the packet transmission into “cycles” and computes the number of bits transmitted through the channel during each cycle, thus as calls are added or removed, the number of bits of the packet that are transmitted during the cycle can change. This models the available bandwidth that varies while the packet is in transmission.

The second algorithm uses the cycle time specified in the “Cycle Time (ms)” field. An appropriate value for many kinds of links is 0.125 milliseconds representing the time between voice samples when sampled at 8000 samples per second.

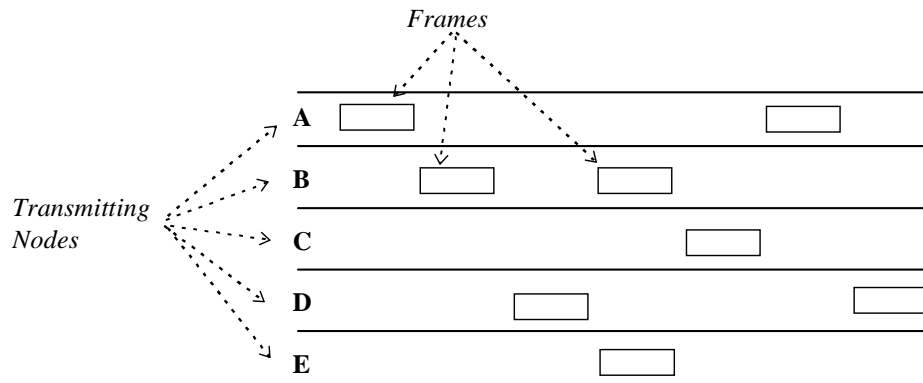
The “Model cycles” list box allows the selection of which algorithm is used. The “Scale entire frame once” option selects the first algorithm, and the “Always” option selects the second option. There is a third option “Call utilization > 0.0” that uses the first algorithm if the call utilization is zero or uses the second algorithm if the utilization is greater than zero. This third option gains simulation efficiency by assuming that packets will transmit so quickly when they have the full bandwidth that the cycles do not need to be modeled.

## 4.39 Link: Aloha

### *Purpose*

An ALOHA link is a multiaccess link to which several nodes may be connected. It is aimed at modeling random access radio links where random transmissions may occur, collisions are detected after transmission, and retransmission scheduled.

Both slotted and unslotted operation may be modeled.



### **Random Transmission From Each Node**

One of the nodes may be designated as a control node. This models the situation where there is a base station or HQ station to which communication is directed by the other users over the radio link. In ALOHA the users would see contention as they broadcast at the same frequency, but the control node does not see contention as it broadcasts at a different frequency back to the users and it only broadcasts one packet at a time.

### *Creating:*

To create an ALOHA link pick any of the link tools from the palette and drag a link onto the background. Then set the type of the link to ALOHA. Or use the create menu option to create an ALOHA link. Once the link has been created a default parameter set is selected. To change the parameter set, select one from the Parameter pulldown box.

*(See Parameter Sets)*

### *Connectivity:*

Multiple nodes can connect to an ALOHA link. One of the connected nodes may be designated as a control node, if the parameters indicate a control channel is available.

Circuit-switched traffic cannot route over an ALOHA link.

### *Editing:*

Double click on the icon to bring up the link dialog box, or highlight the icon and pick the **Edit/Detail** menu option. The name of the link, its icon, and its failure characteristics may then be defined.

Also a parameter set must be applied to the link. The parameter set may be chosen from the model's list. The values found in the parameter set describe the per-

## 4. Modeling Constructs

formance characteristics of the link.  
(See *Parameter Sets*)

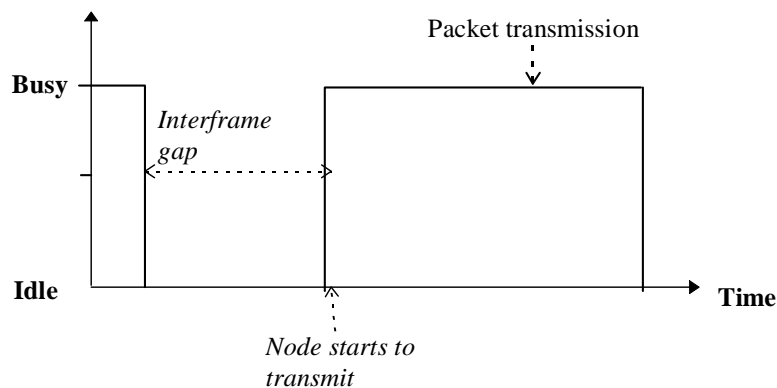
Parameter set name	DEFAULT	
Bandwidth (kbps)	9.6	Retry interval (ms)
Slot width (ms)	0.00000	<input checked="" type="radio"/> Probability distribution
		<input type="radio"/> IEEE binary exponential backoff
Propagation (ms)	0.0000	Retry dist (ms)
Session limit	1024	Exp(1000.0.1)
Frame min (bytes)	0	IEEE binary exponential backoff
Frame max (bytes)	0	Slot time (ms)
Frame OH (bytes)	0	0.05120
<input type="checkbox"/> Frame assembly		Offset (ms)
Frame error prob	0.0000000	0.00000
<input checked="" type="checkbox"/> Automatically retransmit frame errors		Retry limit
		16
		Limit delay (ms)
		1000.0
		Stream
		1
		<input type="checkbox"/> Control channel
		Trans return (kbps)
		9.6

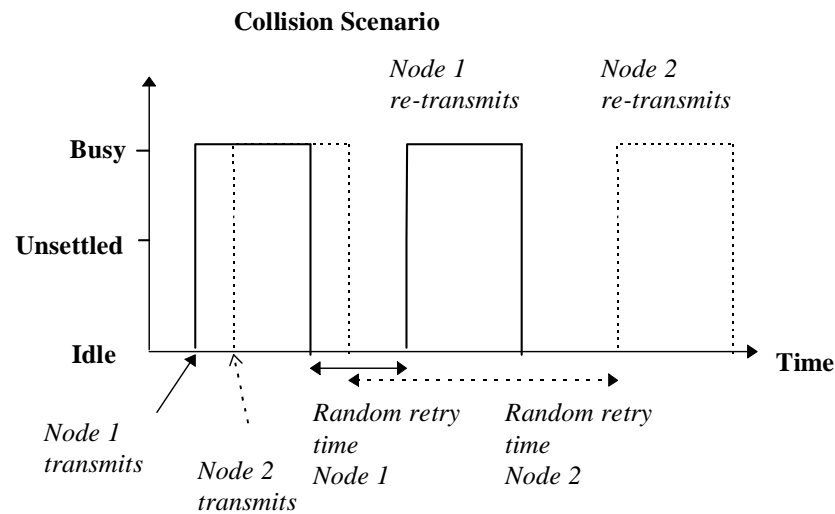
**Aloha Protocol Parameter Set Dialog Box**

### *Execution:*

Multiple nodes are typically connected to an Aloha link. For each node there is an input and output port which connects the node to the link. When a node has data to send over the link it routes the respective packet into the output port buffer where the packet waits until it is its turn for transmission. The packets are ordered in decreasing priority. Packets of the same priority are ordered in a FIFO manner. Consequently the packet at the head of the output port buffer on a node is the packet with the highest priority that has been waiting the longest.

### **Successful Transmission**





When a packet reaches the head of the buffer, the link framing characteristics are inspected and the number of frames required for packet transmission calculated. This will result in 0 or more full frames and 1 partial frame (unless the packet size is exactly a whole number of frames).

The transmission time for the first frame will then be calculated ( $\text{Frame Size} / \text{Transmission Speed}$ ). If a frame error probability is specified then a statistical pick is made to determine whether the frame would arrive in error. If so another transmission time is added and another statistical pick made. This process is repeated until a transmission time for the frame is arrived at which delivers the frame without error. The Aloha link is made busy for this period of time. After the transmission time has elapsed, the link is then available for other transmissions, but the frame must then incur the propagation delay. After the propagation delay, the frame is then at the receiving node.

If no other node has started transmission during the frame transmission time, then no collision has occurred and the next frame is transmitted in the same way. When all frames have arrived at the destination node the original packet is reassembled and placed in the input buffer at the receiving node.

If another node has attempted transmission while a frame is already in transmission then a collision occurs. The link is counted as busy for the full transmission time of both frames. Both frame transmissions are attempted at a later time determined by a random value pulled from the retry distribution. Note that more than 2 frames may collide in the same collision episode. Also, it is possible that another collision will occur on the retransmission attempt.

If the packet has a multicast destination list then the frames that constitute the packet transmission will be transmitted once as described above. At the point of packet reassembly cloned copies of the original packet are placed in the input port buffers of each of the receiving nodes. In other words, a packet with a multicast destination will only be sent once over an Aloha link, but clones of the packet will be placed in the input port buffers of the receiving nodes.

#### 4. Modeling Constructs

If slotted operation is being used then the frame transmissions at each node are always attempted on a slot boundary. If two frames begin transmission at different slot boundaries, then they will not collide, given properly chosen frame sizes and slot times. This has the effect of lessening the total amount of time lost to the collision episodes and improves the throughput performance of the link.

**Reporting:**

Link Delays & Utilization Report  
Random Access Link Performance Report

**Fields:**

**Name** An alphanumeric field to identify the link. The name must be unique in the backbone or subnetwork.

**Icon** The name of the icon used to represent the link.

**Control Node** The name of the node which is the control node for the ALOHA link. A control node has an independent, contention free channel to communicate with the other nodes on the link. The control node is deselected unless the control channel check box is ticked on the link parameters screen.

**Time To Failure** (*see Link*)

**Time To Repair** (*see Link*)

**Time Of Next State Change** (*see Link*)

**Current State** (*see Link*)

**Statistics** Contention and control channel utilization.  
(*see Statistics: Link*)

**Parameter Set Fields:**

**Bandwidth** The speed of the link in kilobits per second.

**Slot Width** If no slot time is entered, COMNET III models unslotted operation. This means that any node may attempt to transmit at any time. If a collision occurs, the transmission continues until the end, the collision is detected, and a retransmission will then be attempted after the retry time.

If a slot time is entered, a node will only transmit on a slot time boundary. If two frames begin transmission at different slot boundaries, then they will not collide, given properly chosen frame sizes and slot times.

**Propagation Delay** The time to propagate a signal across a link. This is generally not significant for terrestrial radio links. For satellite links to geostationary satellites the propagation delay is of the order of 270-300 milliseconds. Propagation delay does not affect link utilization statistics.

**Session Limit** The maximum number of sessions that can be concurrently routed across the link. If a session setup packet attempts to route across a link which is already at its session limit, the routing attempt fails. The alternate routes (if any) are then



tested. If there is still no route the session setup packet is blocked. The Session Limit only applies if connection-oriented routing is used.

**Framing Characteristics**

The framing characteristics for the link. Packets are broken down into transmission frames which are then modeled as being transmitted across the link.  
(see *Link:Framing* )

**Retry Interval**

On the ALOHA protocol, if two or more nodes have overlapping transmissions then a collision occurs. In this event, the transmitting nodes will each pick a random retry time and attempt transmission again. The retry time can be entered as either:

**Probability Distribution**

Use any of the standard COMNET III distributions.  
(see *Distributions: System*)

**Binary Exponential Backoff**

This algorithm is defined by the IEEE for use in collision detecting protocols—particularly CSMA/CD. It may also be picked for the ALOHA retry interval.  
(see *Link: Binary Exponential Backoff*)

**Control Channel**

Check box to indicate whether there is a control channel on this link.

**Transmission Return Rate**

If there is a control channel, what is its speed for transmission between the control node and the other nodes on the link.

## 4. Modeling Constructs

### 4.40 Link: CSMA

#### *Purpose*

A CSMA (Carrier Sense Multiple Access) link is a multiaccess link to which several nodes may be connected. It is aimed at modelling random access links where random transmissions may occur, but only if the link is detected as idle (or unsettled). Like, in Aloha, collisions are detected, and retransmissions will be scheduled.

CSMA differs from Aloha in that a node may detect that a link is busy and defer its transmission until the link is idle. With Aloha, the transmission occurs irrespective of whether the link is idle or not.

CSMA differs from CSMA/CD in that collisions are detected at the end of transmission rather than at the instant when the collision occurs. CSMA does not really do collision detection; it relies on timeouts to indicate that a frame was not delivered (for whatever reason) and schedules retransmission after a random delay. This means that there is wasted bandwidth on CSMA as the transmission duration is effectively lost.

With 1-persistent CSMA, each station transmits with a probability of 1 whenever it finds the link idle. The stations continuously sense the carrier when it is busy and start to send as soon as it is idle. On collision, it backs off for a random time period.

With p-persistent CSMA, a station transmits only with a certain probability. If the channel is found to be idle, a random number is generated. If this is greater than p, then the station defers the transmission until the next slot. If the random number is less than or equal to p, the station starts transmission.

With non-persistent CSMA, a station does not keep on sensing the channel until it is idle. If it finds the link busy, it waits a random interval without sensing and then senses again after this interval has elapsed.

COMNET III only models 1-persistent CSMA.

One of the nodes may be designated as a control node. This node may then have a contention free channel to communicate with the other nodes on the link.

#### *Creating:*

To create a CSMA link pick the appropriate link tool from the palette and drag a link onto the background. Or use the create menu option to create a CSMA link. Double clicking on the icon brings up a dialog box where you can indicate in the type box to use CSMA operation. Once the link has been created a parameter set must be applied to it via a selection in the Parameter pulldown box.  
(See *Parameter Sets*)

#### *Connectivity:*

Multiple nodes can connect to a CSMA link. One of the connected nodes may be designated as a control node.

Circuit-switched traffic cannot route over a CSMA link.

#### *Editing:*

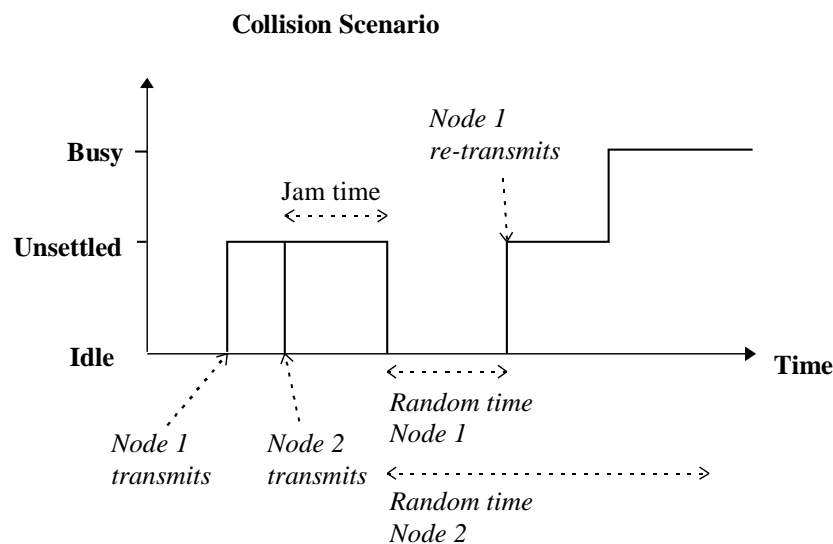
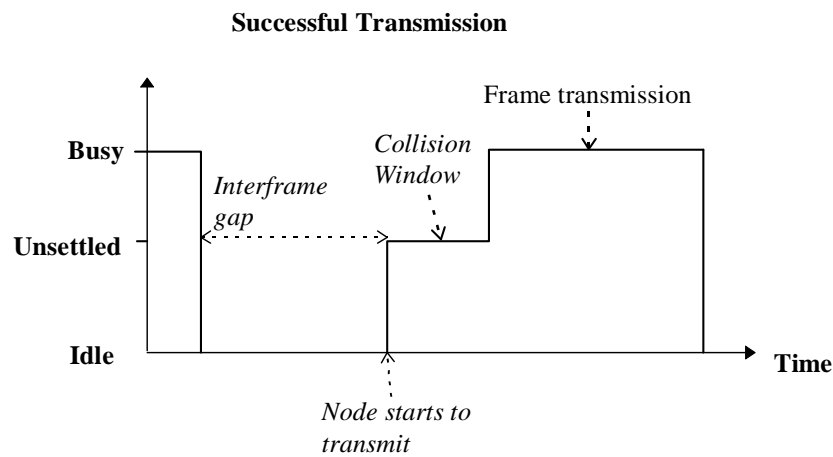
Double click on the icon to bring up the link dialog box, or highlight the icon and pick the *Edit/Detail* menu option. The name of the link, its icon, and its failure characteristics may then be defined.

Also a parameter set must be applied to the link. The parameter set may be chosen from the model's list. The values found in the parameter set describe the performance characteristics of the link.

(See *Parameter Sets*)

**Execution:**

Multiple nodes are typically connected to a CSMA link. For each node there is an input and output port which connects the node to the link. When a node has data to send over the link it routes the respective packet into the output port buffer where the packet waits until it is its turn for transmission. The packets are ordered in decreasing priority. Packets of the same priority are ordered in a FIFO manner. Consequently the packet at the head of the output port buffer on a node is the packet with the highest priority that has been waiting the longest.



When a packet reaches the head of the buffer, the link framing characteristics are

## 4. Modeling Constructs

inspected and the number of frames required for packet transmission calculated. This will result in 0 or more full frames and 1 partial frame (unless the packet size is exactly a whole number of frames).

If the link is busy, the transmission attempt will be deferred until the link becomes idle, plus a time delay equal to the interframe gap. If the link is idle then a transmission attempt can be made immediately.

The link is then placed in an unsettled state for the collision window period. The transmission time for the frame will be calculated ( $\text{Frame Size}/\text{Transmission Speed}$ ). If a frame error probability is specified then a statistical pick is made to determine whether the frame would arrive in error. If so another transmission time is added and another statistical pick made. This process is repeated until a transmission time for the frame is arrived at which delivers the frame without error. The link is then made busy for the calculated transmission time, after which it is made idle. The link is then available for other transmissions, but the frame must then incur the propagation delay. After the propagation delay the frame is then at the receiving node.

When all frames have arrived at the receiving node the original packet is reassembled and placed in the input buffer at the receiving node.

The most likely collision scenario occurs when two or more nodes sense the link is busy and wait until it becomes idle to start transmitting. Since two or more are trying to transmit at the same time, collisions result.

Another, less likely, collision scenario occurs if another node has attempted transmission during the collision window while the CSMA link is in an unsettled state. Both nodes keep transmitting for the calculated transmission time. Then the link may be jammed for the optional jam time to alert all stations that a collision episode has occurred. A random pull is then made for each frame from the retry distribution and both frame transmissions attempted later. Note that more than 2 frames may collide in the same collision episode. Also there is no guarantee that another collision will not reoccur on the retransmission attempt.

If the packet has a multicast destination list then the frames that constitute the packet transmission will be transmitted once as described above. At the point of packet reassembly cloned copies of the original packet are placed in the input port buffers of each of the receiving nodes. In other words, a packet with a multicast destination will only be sent once over a CSMA link, but clones of the packet will be placed in the input port buffers of the receiving nodes.

### ***Reporting:***

Link Delays & Utilization Report  
Random Access Link Performance Report

### ***Fields:***

<b>Name</b>	An alphanumeric field to identify the link. The name must be unique in the backbone or subnetwork.
<b>Icon</b>	The name of the icon used to represent the link.
<b>Control Node</b>	The name of the node which is the control node for the CSMA link. A control

node has an independent, contention free channel to communicate with the other nodes on the link. The control node is deselected unless the control channel check box is ticked on the link parameters screen.

**Time To Failure** (see *Link*)

**Time To Repair** (see *Link*)

**Time Of Next State Change** (see *Link*)

**Current State** (see *Link*)

**Statistics** Contention and control channel utilization.  
(see *Statistics: Link*)

### ***Parameter Set Fields:***

**Bandwidth** The speed of the link in kilobits per second.

**Collision Window** The period of time that the link is vulnerable to collision after a node begins transmitting.

**Jam Interval** With CSMA, after a node recognizes a collision at the end of a frame transmission, it may send a jamming signal to alert other nodes of collision for the jam interval. After a collision, the link is not idle again until the jam time has expired.

With CSMA the jam time is used to model the interval from the end of the transmission until the sending node learns that retransmission is required.

**Interframe Gap** The duration of an interframe gap which is the length of time a node delays its transmission attempt after a link becomes idle. The node may be transmitting a number of frames sequentially—the interframe gap is applied between each frame followed by recontention for the link. Alternatively, the node may be waiting for a transmission by another node to finish—again it will wait the interframe gap after transmission completes before contending for the link.

**Propagation Delay** The time to propagate a signal across a link.

**Session Limit** If connection-oriented routing is used, this is the maximum number of sessions that can be concurrently routed across the link. If a session setup packet attempts to route across a link which is already at its session limit, the routing attempt fails. The alternate routes (if any) are then tested. If there is still no route the session setup packet is blocked.

**Framing Characteristics** The framing characteristics for the link. Packets are broken down into transmission frames which are then modeled as being transmitted across the link.  
(see *Link:Framing* )

**Retry Interval** On the CSMA protocol, if two or more nodes have overlapping transmissions then a collision occurs. In this event the transmitting nodes will each wait the jam time followed by a random retry time and attempt transmission again. The retry time can be entered as either:

#### 4. Modeling Constructs

##### **Probability Distribution**

Use any of the standard COMNET III distributions.  
(see *Distributions*)

##### **Binary Exponential Backoff**

This algorithm is defined by the IEEE for use in 802.3 collision detecting protocols.

(see *Link: Binary Exponential Backoff*)

##### **Control Channel**

Check box to indicate whether there is a control channel on this link.

##### **Transmission Return Rate**

If there is a control channel, specifies its speed in kilobits per second.

## 4.41 Link: CSMA/CA Wireless LAN IEEE 802.11

### **Purpose:**

The CSMA/CA model is a shared medium access mechanism based on the Distributed Convergence Function (DCF) of the IEEE 802.11 Wireless LAN specification. The basic procedure for accessing the shared medium is as follows.

Each station wishing to transmit first senses the medium to determine if another station is transmitting. If the medium is not busy, the transmission may proceed. The IEEE802.11 distributed algorithm mandates that a gap of a minimum specified duration (called the Distributed Inter-Frame Space, or DIFS) exists between contiguous frame transmissions. A transmitting station ensures that the medium is idle for this DIFS duration before attempting to transmit. If the medium is sensed busy, the station defers until the end of the current transmission. After deferral, or prior to attempting to transmit again immediately after a successful transmission, the station selects a random backoff interval and decrements the backoff interval counter while the medium is free. When the backoff interval counter reaches zero, the station proceeds to transmit the data. The CSMA/CA protocol is designed to reduce rather than eliminate the collision probability between multiple stations accessing a medium at the point where collisions would most likely occur. Just after the medium becomes free following a busy period is when the highest probability of a collision occurs. Therefore a random backoff arrangement is immediately applied in this situation. In addition, all directed (non-multicast) traffic uses immediate positive acknowledgments (ACK frames). Retransmission is scheduled by the sender if no ACK is received.

### **Bandwidth:**

This parameter defines the bit rate supported by the 802.11 wireless LAN physical medium. The supported data rates in the IEEE standard are either 1 Mbps or 2 Mbps (but COMNET III does not limit you to these values).

### **Propagation:**

This parameter specifies the air propagation time it takes a transmitted signal to go from the transmitting station to the receiving station. A nominal value of 1 microsecond has been allocated for this parameter in the IEEE 802.11.

### **Slot time:**

The parameter *Slot time* is used to define the *Short* and *Distributed Interframe Spaces* and to update the *backoff interval* in the shared medium access algorithm. A nominal value of 6 microseconds is specified for the infrared physical system. *Users shall specify this value according to their manufacturer's product specification.*

### **Short IFS:**

The *Short Interframe Space* is the shortest of the interframe spaces defined in IEEE 802.11. It is used to provide an efficient MAC Service Data Unit (MSDU) delivery mechanism. Once the medium is seized, the station will keep it for the duration of the frame exchange it has to perform. In the implemented model, the *Short IFS* is used between fragments of one MSDU and their ACK frames. Using the smallest gap between transmissions within the frame exchange prevents other stations, which are required to wait for the medium to be free for a longer gap (the *Distributed IFS*, see below) from attempting to use the medium, thus giving priority to completion of the frame exchange in progress.

A nominal value of 7 microseconds is specified for infrared physical system. *User shall specify this parameter based on manufacturer's product specification.*

The *Short IFS (SIFS)* is also used to compute the *Distributed IFS (DIFS)* based on the following equation:

## 4. Modeling Constructs

$$\text{Distributed IFS} = \text{Short IFS} + 2 * \text{Slot Time}$$

The *DIFS* is used by the CSMA/CA model to contend for the channel for initial frame transmissions. A station is allowed to transmit if it detects the medium to be free for the *DIFS* duration and its backoff time has expired.

**Frame max:** This parameter specifies the maximum MSDU length that will be accepted for transmission. The default value is 2304 octets.

**Frame OH:** The frame overhead parameter includes the 802.11 MAC header, an IEEE 32-bit CRC, and 8 octets of shared key for authentication. In the case of fragmentation (see below), this is also the overhead of transmitting a fragment. The default value of the frame overhead is 42 octets.

**Frgmt thrshld:** The fragmentation threshold specifies the current maximum size, in octets, of the Mac Protocol Data Unit (MPDU) that will be delivered to the physical layer. The default value is 2304, which is equal to the maximum frame size. Fragmentation creates MPDUs smaller than the MSDU (frame) size to increase reliability by increasing the probability of successful transmission of the MSDU (frame) in cases where channel characteristics limit transmission reliability for longer frames. When a frame is received with MSDU size greater than the specified "Fragmentation Threshold," the frame must be fragmented. The MPDUs (or fragments) resulting from the fragmentation of an MSDU are sent as independent transmissions, each of which is separately acknowledged. This permits transmission retries to occur per fragment, rather than per MSDU (frame).

Unless interrupted due to medium occupancy limitations for a given physical system, such as a dwell time boundary in a Frequency Hopping Spread Spectrum (FHSS) system, the fragments of a single MSDU are sent as a burst, using a single invocation of the CSMA/CA medium access procedure. In other words, once the station has contended for the channel, it will continue to send fragments (by using the *Short Interframe Space* instead of the *Distributed Interframe Space*) until either all fragments of a MSDU have been sent, an acknowledgment is not received for directed data due to collision or transmission error, or the station can not send any additional fragments due to a dwell time boundary. Should the sending of the fragments be interrupted due to one of these reasons, the frame or the fragment will enter a retransmission backoff period; when the next opportunity for transmission occurs the station resumes sending the rest of the fragments of the frame.

**Frgmt error prob:** This parameter specifies the error rate of transmitting one fragment (or frame, in case of no fragmentation) over the physical medium.

**ACK timeout:** This parameter specifies the length of time by which an ACK frame should be received in response to transmission of a frame which requires acknowledgment. It is timed from the instant that the transmitting station finishes the transmission of the frame or fragment.

The reception of an unicast frame (or fragment in the case where fragmentation is needed) requires the receiving station to respond with an ACK frame, if the received frame (fragment) is correct. Lack of reception of an expected ACK frame indicates to the source station that an error has occurred and causes the station to retransmit the frame (or fragment). An erroneous ACK frame causes the retransmission of the transmitted frame (or fragment) as well. The transmission



of the ACK frame commences after the *Short IFS* without regard to the busy/free state of the medium.

The *ACK Timeout* may not be less than  $(2 * Propagation + Short IFS + Ack Transmission Time)$ . Otherwise an error message will occur in the dialog box.

**ACK error prob:**

This parameter specifies the error rate of transmitting an ACK frame over the physical medium. An ACK frame is 14 octets long.

**Frame TxLifeTime:**

The frame transmission life time parameter is defined as the elapsed time, after initial transmission of an MSDU, after which further attempts to transmit the MSDU will be terminated. Therefore, this parameter is per frame and not per fragment. The default value is 512 milliseconds.

**FH dwell time:**

The FH dwell time parameter is specified for an FHSS physical system. The attribute is defined as the time the physical medium dependent system can dwell on an FH channel and meet the requirements of the current regulatory domain.

In the implemented CSMA/CA model, the FH dwell time is used for determining if the station can retain the control of the channel during the process of transmitting fragments of an MSDU. Once the dwell boundary is up, the frame has to enter the backoff period and contend again for the channel.

**IEEE 802.11 Backoff:**

This part of the dialog box provides a group of parameters for the backoff algorithm. A station that desires to initiate transfer of data and finds the medium busy follows the backoff procedure. To begin the backoff procedure, the station selects a backoff time based on the following equations.

$$\text{Backoff Time} = \text{INT}(\text{CW} * \text{Random}()) * \text{Slot Time}$$

where:

*CW* = An integer between the values of *CW min* and *CW max*

*Random()* = Pseudo-random number between 0 and 1

*Slot Time* = the value specified above

The *Contention Window (CW)* parameter takes an initial value of the *CW min* for every MPDU queued for transmission. The *CW* will take the next value in the series at every retry to send a particular MPDU until it reaches the value of the *CW max*. The set of *CW* values are 7 (*CW min*), 15, 31, 63, 127, 255 (*CW max*).

The parameter “*Retry limit*” indicates the maximum number of retransmission attempts of a fragment. It is set as the default value 7. Once this limit is reached, a failure condition is indicated to the next higher layer. Note that the statistics report on “Number of tries to resolve” may have a bigger value than this parameter. This is because the statistics field is on frame basis while the retry limit in the implemented CSMA/CA is on fragment basis.

All backoff periods occur following a DIFS period during which the medium is free. A station in backoff monitors the medium for carrier activity during backoff periods. If no carrier activity is seen for the duration of a particular slot, then the random backoff process decrements the backoff timer by *Slot time* specified above. If there is carrier activity sensed at any time during a backoff period, then the backoff procedure is suspended; that is, the backoff timer will not be decre-

#### 4. Modeling Constructs

ment for that slot. The medium must be sensed as idle again for the duration of a DIFS before the backoff procedure is allowed to resume. Transmission commences whenever the backoff timer reaches zero.

A station that had just transmitted an MSDU and has another MSDU ready to transmit will perform the backoff procedure in order to produce a level of fairness of access to the medium amongst stations.

Some of the performance measures on the output reports have special interpretations for CSMA/CA links. On the Channel Utilization report, the column for FRAMES RST/ERR shows a count of the number of frames that reach the Retry Limit or the XmitLifeTime. On the same report, utilization does not include the time required for transmissions in error. On the Collision Stats report, NBR OF TRIES TO RESOLVE includes retransmissions due to both collisions and errors.

## 4.42 Link: CSMA/CD

### *Purpose*

A CSMA/CD (Carrier Sense Multiple Access With Collision Detection) link is a multiaccess link to which several nodes may be connected. It is aimed at modelling random access links where random transmissions may occur, collisions are detected, and retransmission scheduled. The protocol is modeled on the IEEE 802.3 standards and, as such, is suitable for modelling a wide variety of Ethernet based protocols.

A number of CSMA/CD implementations are contained in the system library, the details of which are outlined in the table below.

**Table 1: CSMA/CD Implementation Details**

	CSMA/ CD10Base5	CSMA/ CD10Base2	CSMA/ CD1Base5	CSMA/ CD1Base5 Star	CSMA/ CD10BaseT	TOP	Ethernet 10Base5	CSMA/ CD100BaseT
Media	Coax 50 Ohm	Coax 50 Ohm	Unshielded Twisted Pair	Unshielded Twisted Pair	Unshielded Twisted Pair		Coax50 Ohm	
Diameter mm	10	5	0.4-0.6	0.4-0.6	0.4-0.6		10	
Max segment length m	500	185	500	500	100		500	
Max length m	2500	925	2500	2500	500		2500	
Transmission speed km/ sec	231000	195000	177000	177000	177000		231000	
Max propagation delay ms.	0108225	0.0047358	.01412420	.	00282485		.0108225	
Nodes per segment	100	30	-		-		100	
Data Rate Mbps	10	10	1	1	10	10	10	
Collision Window ms	.01	.0018	.14	.1	.002	.01	.01	.002
Jam Time ms	.0032	.0032	.032	.032	.0032	.0032	.0032	.0032
Interframe Gap ms	.0096	.0096	.096	.096	.0096	.0096	.0096	.0096
Propagation Delay								
Session Limit	1024	1024	1024	1024	1024	1024	1024	1024
Frame Min	72	72	72	72	72	72	72	72
Frame Max	1526	1526	1526	1526	1526	1526	1526	1526
Frame Overhead	30	30	30	29	30	29	26	30
Slot Time ms	.0512	.0512	.0512	.0512	.0512	.0512	.0512	.00512
Offset ms	0	0	0	0	0	0	0	0
Retry Limit	16	16	16	16	16	16	16	16
Limit Delay ms	1000	1000	1000	1000	1000	1000	1000	1000

One of the nodes may be designated as a control node. This node may then have a contention free channel to communicate with the other nodes on the link.

### *Creating:*

To create a CSMA/CD link pick the appropriate link tool from the palette and drag a link onto the background. Or use the create menu option to create a CSMA/CD link. On the dialog box accessed via double clicking on the icon, you can indicate in the type box whether to use CSMA or CSMA/CD operation. Once

#### 4. Modeling Constructs

the link has been created a parameter set must be applied to it via a selection in the Parameter pulldown box.  
(See *Parameter Sets*)

##### **Connectivity:**

Multiple nodes can connect to a CSMA/CD link. One of the connected nodes may be designated as a control node.

Circuit-switched traffic cannot route over a CSMA/CD link.

##### **Editing:**

Double click on the icon to bring up the link dialog box, or highlight the icon and pick the **Edit/Detail** menu option. The name of the link, its icon, and its failure characteristics may then be defined.

Also a parameter set must be applied to the link. The parameter set may be chosen from the model's list. The values found in the parameter set describe the performance characteristics of the link.  
(See *Parameter Sets*)

##### **Execution:**

Multiple nodes are typically connected to a CSMA/CD link. For each node there is an input and output port which connects the node to the link. When a node has data to send over the link it routes the respective packet into the output port buffer where the packet waits until it is its turn for transmission. The packets are ordered in decreasing priority. Packets of the same priority are ordered in a FIFO manner. Consequently the packet at the head of the output port buffer on a node is the packet with the highest priority that has been waiting the longest.

When a packet reaches the head of the buffer the link framing characteristics are inspected and the number of frames required for packet transmission calculated. This will result in 0 or more full frames and 1 partial frame (unless the packet size is exactly a whole number of frames).

If the link is busy, the transmission attempt will be deferred until the link becomes idle, plus a time delay equal to the interframe gap. If the link is idle then a transmission attempt can be made immediately.

The link is then placed in an unsettled state for the collision window period. Assuming the no collisions have occurred in the collision window the transmission time for the frame will be calculated ( $\text{Frame Size}/\text{Transmission Speed}$ ). If a frame error probability is specified then a statistical pick is made to determine whether the frame would arrive in error. If so another transmission time is added and another statistical pick made. This process is repeated until a transmission time for the frame is arrived at which delivers the frame without error. The link is then made busy for the calculated transmission time, after which it is made idle. The link is then available for other transmissions, but the frame must then incur the propagation delay. After the propagation delay the frame is then at the receiving node.

When all frames have arrived at the receiving node the original packet is reassembled and placed in the input buffer at the receiving node.

If another node has attempted transmission during the collision window while the CSMA/CD link is in an unsettled state, a collision occurs. The link is jammed for the jam time to alert all stations that a collision episode has occurred. A random pull is then made for each frame from the retry distribution and both frame trans-

missions are attempted later. Note that more than 2 frames may collide in the same collision episode. Also there is no guarantee that another collision will not reoccur on the retransmission attempt.

If the packet has a multicast destination list then the frames that constitute the packet transmission will be transmitted once as described above. At the point of packet reassembly cloned copies of the original packet are placed in the input port buffers of each of the receiving nodes. In other words, a packet with a multicast destination will only be sent once over a CSMA/CD link, but clones of the packet will be placed in the input port buffers of the receiving nodes.

**Reporting:**

Link Delays & Utilization Report  
Random Access Link Performance Report

**Fields:**

<b>Name</b>	An alphanumeric field to identify the link. The name must be unique in the backbone or subnetwork.
<b>Icon</b>	The name of the icon used to represent the link.
<b>Control Node</b>	The name of the node which is the control node for the CSMA/CD link. A control node has an independent, contention free channel to communicate with the other nodes on the link. The control node is deselected unless the control channel check box is ticked on the link parameters screen.
<b>Time To Failure</b>	<i>(see Link)</i>
<b>Time To Repair</b>	<i>(see Link)</i>
<b>Time Of Next State Change</b>	<i>(see Link)</i>
<b>Current State</b>	<i>(see Link)</i>
<b>Statistics</b>	Contention and control channel utilization. <i>(see Statistics: Link)</i>
<b>Parameter Set Fields:</b>	
<b>Bandwidth</b>	The speed of the link in kilobits per second.
<b>Collision Window</b>	The period of time that the link is vulnerable to collision after a node begins transmitting. On a 2500 meter cable at 4.33 microseconds per kilometer, this would be approximately .01 milliseconds. The worst case would be the two way transmission time between nodes at the opposite end of the cable—the average case for 2 arbitrary nodes would be half the worst case. This window may be adjusted for different conditions: a smaller window for short connections and a longer interval for long connections.
<b>Jam Interval</b>	With CSMA/CD, after a node recognizes a collision it sends a jamming signal to alert other nodes of the collision for the jam interval. After a collision the link is not idle again until the jam time has expired.

## 4. Modeling Constructs

In contrast, CSMA models the jam time as the interval from the end of the transmission until the sending node learns that retransmission is required.

### Interframe Gap

The duration of an interframe gap which is the length of time a node delays its transmission attempt after a link becomes idle. The node may be transmitting a number of frames sequentially—the interframe gap is applied between each frame followed by recontention for the link. Alternatively, the node may be waiting for a transmission by another node to finish—again it will wait the interframe gap after transmission completes before contending for the link.

The default value for a 10MB IEEE 802.3 LAN is .0096 milliseconds. This gap provides recovery time for other Ethernet stations and provides opportunity for other stations to acquire the link by colliding with (interrupting) a station's long sequence of frames.

### Propagation Delay

The time to propagate a signal across a link.

### Session Limit

If connection-oriented routing is used, then this is the maximum number of sessions that can be concurrently routed across the link. If a session setup packet attempts to route across a link which is already at its session limit, the routing attempt fails. The alternate routes (if any) are then tested. If there is still no route the session setup packet is blocked.

### Framing Characteristics

The framing characteristics for the link. Packets are broken down into transmission frames which are then modeled as being transmitted across the link. (*see Framing Characteristics*)

### Retry Interval

On the CSMA/CD protocol, if two or more nodes have overlapping transmissions (i.e., two or more transmission attempts within the collision window) then a collision occurs. In this event the transmitting nodes will each wait the jam time followed by a random retry time and attempt transmission again. The retry time can be entered as either:

### Probability Distribution

Use any of the standard COMNET III distributions. (*see Distributions*)

### Binary Exponential Backoff

This algorithm is defined by the IEEE for use in 802.3 collision detecting protocols. (*see Link: Binary Exponential Backoff*)

### Control Channel

Check box to indicate whether there is a control channel on this link.

### Transmission Return Rate

If there is a control channel, specifies its speed in kilobits per second.

#### 4.43 Link: Demand-Assigned Multiple Access (DAMA)

*Purpose:*

The Demand-Assigned Multiple Access (DAMA) link allows multiple nodes to share a pool of circuits. The link has the same set of parameters as a point-to-point link and can carry both packets and circuit-switched calls. As with the point-to-point link, the DAMA link allows calls and packets to share the same bandwidth using the call-based link loading option. Each node connected to the DAMA link can access only 1 circuit of the DAMA link's pool of circuits at a time. A circuit is allocated to a node for the amount of time required to transmit a packet. When several nodes are waiting to transmit packets, the node with the earliest arriving packet uses the first circuit to become available.

## 4. Modeling Constructs

### 4.44 Link: FDDI and Priority FDDI

*Purpose:*

COMNET III includes two models of FDDI--a Basic FDDI model and a Priority FDDI model. As with the token-passing link, the COMNET III FDDI models do not actually schedule each token-passing event; instead, the model keeps track of the last node to have relinquished the token and the time when the node relinquished the token. Given the last token location and the time that the frame left that location, COMNET III computes the appropriate amount of time (token passing delay) required for the token to advance the next time some node has a frame to transmit. COMNET III monitors the actual token rotation time and the token-ring report includes a count of the number of times that the actual token rotation time exceeds twice the target token rotation time to help identify heavily loaded conditions that may trigger FDDI re-initialization.

The Basic FDDI model has a Token-Passing Delay that determines the time required to pass the token from one node on the FDDI's connection list to the next node on the connection list. The time that a node can hold the token is controlled through the Target Token Rotation Time (TTRT). When a node receives the token, before transmitting a frame it determines the time that has elapsed since the node last received the token. If the elapsed time exceeds the TTRT, then the node passes the token to the next node without transmitting the waiting frame. The node can continue to transmit frames as long as the elapsed time since last receiving the token does not exceed the TTRT.

The Priority FDDI model allows the TTRT to vary by frame priority which is the highest priority of the packets it carries. The TTRTs can also vary by port, so that some nodes on the Priority FDDI link can hold the token longer than other nodes. The priority FDDI parameter set maintains a list of tables for different token rotation time targets (T\_Pr) that vary by priority and the different tables are available for assignment to different ports (node interfaces). By default, there is one table called "DEFAULT" and it will be assigned to each new port added to the link.

The Priority FDDI parameter set dialog box has a button "T\_Pr tables" that manages the sets of tables that may be assigned to the individual ports. Use the "T\_Pr tables" button to edit the list of port properties available to be assigned to each port connecting to the priority FDDI link. Each port properties dialog box consists of a table of threshold values for different levels of priority that may be edited as shown in the following dialog. The table shows the lower limit of the frame priority that will use the threshold values in the T\_Pr column. Priorities in COMNET III are such that the larger the number the higher the priority. Generally the threshold token-rotation times should increase for increasing priorities to allow those priorities to use the link longer than the lower priorities.



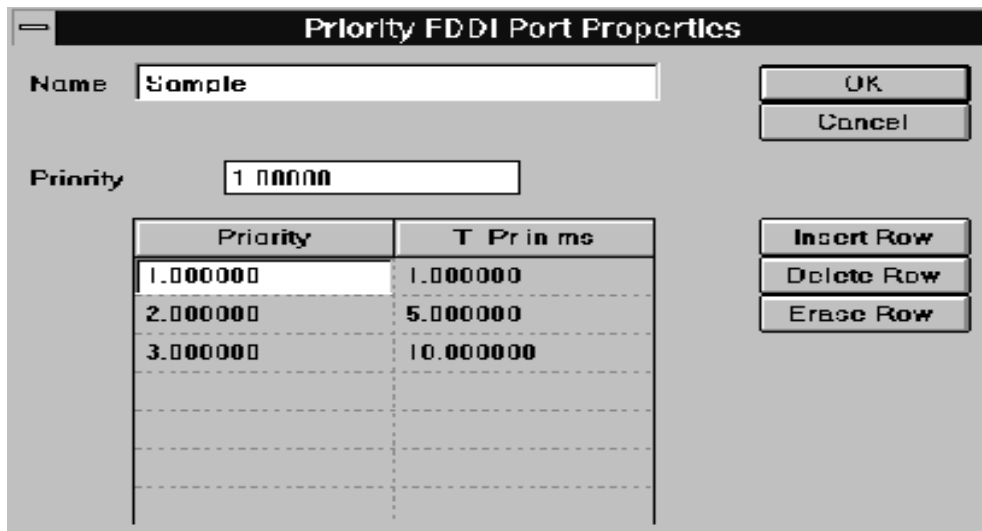


Figure 5. Priority FDDI port properties

The priority FDDI parameter set maintains this list of port properties so that the port properties may be assigned to a set of ports requiring the same parameters. Thus, the port properties dialog reuses data for multiple ports, while the parameter set itself allows the data to be reused for different links.

The port properties are assigned to the various ports through the port properties dialog attached to the arc between the node and the link icons. The bottom field of the port properties dialog allows the assignment of a specific priority-FDDI port parameter set to be assigned to this port. The “..” button next the field will view the selected parameter set, but will not allow editing. Editing the parameter set can only be done through the link’s parameter dialog because these port parameters may be used in multiple places.

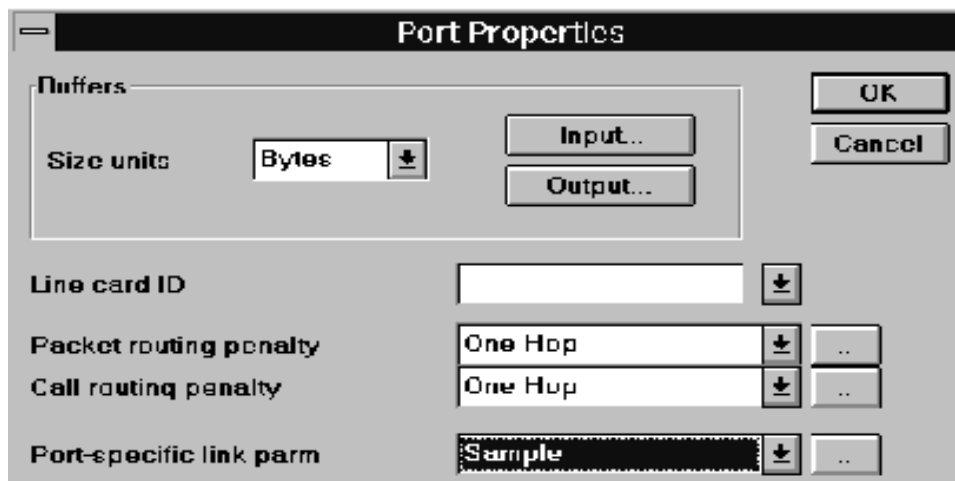


Figure 6. Port properties (on arc between node and priority-FDDI link)

#### 4. Modeling Constructs

The priority FDDI is closely related to the priority scheme used in the IEEE 802.4 standard except that in 802.4 the highest priority frame has permission to transmit for a holding limit instead of a token-rotation threshold time. The Priority FDDI parameter set has an option to use this 802.4 option of priority operation and provides a field for specifying the holding limit value.

## 4.45 Link: Point-To-Point

### *Purpose*

A point-to-point link connects two nodes together. It is aimed at modelling leased lines, serial lines, trunk circuits, and general WAN data paths (such as transatlantic links or satellite links) established between two points.

For data communications the point-to-point link is a full duplex bidirectional communications path that is characterized by its speed and number of channels. Frames are multiplexed onto the link in the order that they appear at the head of the link port transmission buffer.

For voice or circuit-switched traffic, separate bandwidth and/or channels may be defined. Only point to point links may carry circuit-switched traffic on their designated circuit-switched capacity.

### *Creating:*

To create a Point-To-Point link, pick the appropriate link tool from the palette and drag a link onto the background. Or use the create menu option to create a Point-To-Point link. On the dialog box accessed via double clicking on the icon you can indicate in the type box to use Point-To-Point operation. Once the link has been created a parameter set must be applied to it via a selection in the Parameter pulldown box.

*(See Parameter Sets)*

### *Connectivity:*

Only two nodes may connect to a Point-To-Point link. A Computer Group node cannot be connected to a Point-To-Point link.

### *Editing:*

Double click on the icon to bring up the link dialog box, or highlight the icon and pick the **Edit/Detail** menu option. The name of the link, its icon, and its failure characteristics may then be defined.

Also a parameter set must be applied to the link. The parameter set may be copied from the system library, or created locally within the model. The values found in the parameter set describe the performance characteristics of the link.

*See Parameter Sets.*

## 4. Modeling Constructs

	Packets, Frames, Cells	Circuit-Switched Calls
Parameter set name	DEFAULT	
Number of circuits	1	1
Bandwidth/circuit (kbps) (from node X)	1536.000	1536.000
Bandwidth/circuit (kbps) (from node Y)	1536.000	
Bandwidth reserved for 1-hop calls (%)		0.000000
Propagation (ms)	0.000	
Session limit	1024	
Frame min (bytes)	0	
Frame max (bytes)	0	
Frame overhead (bytes)	0	
<input type="checkbox"/> Frame assembly		
Frame error prob	0.0000000	
<input checked="" type="checkbox"/> Automatically retransmit frame errors		

**Point-To-Point Link Parameters Dialog Box**

### *Execution:*

#### **Packet Switching**

As an example, consider two nodes (A & B say) connected to a Point-To-Point link which is modeled as having full-duplex operation. The link may have a number of circuits specified, N, which means that there are N channels for transmission from A to B and another N channels from B to A. For both nodes there is an input port at each node servicing all the circuits on the link sending data to the node, and one output port at each node servicing all the circuits on the link over which transmissions may occur. If a node is connected to several links then there are separate input and output ports for each link.

When a node has data to send over the link it routes the respective packet into the output port buffer where the packet waits until it is its turn for transmission. The packets are ordered in decreasing priority. Packets of the same priority are ordered in a FIFO manner. Consequently the packet at the head of the output port buffer on a node is the packet with the highest priority that has been waiting the longest. A packet must incur the port processing delay before it can be transmitted.

When a packet reaches the head of the buffer, the link framing characteristics are inspected and the number of frames required for packet transmission calculated. This will result in 0 or more full frames and 1 partial frame (unless the packet size is exactly a whole number of frames).

Transmission will start when one of the circuits on the link becomes idle. The transmission time for the first frame will be calculated ( $\text{Frame Size} / \text{Circuit Transmission Speed}$ ). If a frame error probability is specified then a statistical pick is made to determine whether the frame would arrive in error. If so another transmission time is added and another statistical pick made. This process is

repeated until a transmission time for the frame is arrived at which delivers the frame without error. The link is then made busy for the calculated transmission time, after which it is made idle. The link is then available for the next frame transmission, but the frame that has just been transmitted must then incur the propagation delay. After the propagation delay the frame is then at the receiving node.

When all frames have arrived at the receiving node the original packet is reassembled and placed in the input buffer at the receiving node.

No particular distinction is made between one circuit or another on the link. The link object has two integer attributes—the total number of circuits on the link and the number of circuits in use. The link has capacity to start a frame transmission when the number of circuits in use is less than the total number of circuits. When the transmission starts the number of circuits in use is incremented by 1. When the transmission completes the number of circuits in use is decremented by 1. The individual circuits per se are not modeled. If a link fails, all the circuits on the link fail.

#### **Circuit Switching**

A Point-To-Point link may have a circuit-switched bandwidth defined as well as a packet-switch bandwidth. The circuit-switched bandwidth is completely independent of the packet-switched bandwidth. Only circuit-switched traffic originating from Call Sources utilizes the circuit-switched bandwidth.

When a circuit-switched call originates, a routing decision is made from origin to destination. This will result in zero, one, or more links being selected to carry the call. The routing decision is instantaneous in the model and checks (among other things such as hop limits, availability, etc.) that there is sufficient bandwidth on the link to satisfy the call requirement. The required bandwidth is then claimed from the link and held for the duration of the circuit-switched call, after which the call terminates and the bandwidth is given back to the link.

A call routed over a link may terminate early for two reasons. These are:

- 1.. The link (or another node or link on the route) may fail which will mean that the call is disconnected and the bandwidth on all nodes and links over which the call was routed instantaneously relinquished.
- 2.. If preemption is enabled then a higher priority call may preempt a lower priority call on the link. This will cause the low priority call to relinquish its bandwidth across its entire route.

#### **Reporting:**

Link Delays & Utilization Report  
Call Statistics by Link  
Link Utilization Statistics for Calls

Link delays are also inclusive on the Message, Session and Packet delay reports. Buffering on the links is reported in the Buffer reports.

#### **Fields:**

##### **Name**

An alphanumeric field to identify the link. The name must be unique in the back-

## 4. Modeling Constructs

bone or subnetwork.

<b>Icon</b>	The name of the icon used to represent the link.
<b>Control Node</b>	Not supported for point-to-point links.
<b>Link State</b>	Indicates whether the link is in a failed state or not. ( <i>see Link</i> )
<b>Statistics</b>	Various statistics may be collected and graphically displayed for Point-To-Point link utilization. These include Busy Channels From Node 1 and Busy Channels From Node 2 utilization. ( <i>see Statistics: Link</i> )

### ***Parameter Set Fields:***

#### **Point-To-Point Link— Circuit Switching Characteristics**

<b>Number Of Circuits</b>	The number of identical trunk circuits available to carry circuit-switched traffic.
<b>Bandwidth</b>	The bandwidth per circuit. The total capacity for carrying circuit-switched traffic is the number of circuits multiplied by the bandwidth. A trunk could therefore be described as 30 circuits each at 64K, or 1 circuit at 1920K. When a circuit-switched call is routed over the link, the call has a bandwidth requirement. If the current free capacity on the link is greater than the call requirement then the call may be carried. The free capacity is therefore reduced by the bandwidth requirement for the call holding time.

#### **Bandwidth Reserved For 1-Hop Calls**

It is common for directly-routed traffic in a voice network to have bandwidth reserved for it on the trunk. The amount of reserved bandwidth for one-hop calls may be specified. The one-hop calls can originate at either node connected to the point-to-point link. The reserved bandwidth is never available to alternate-routed calls.

#### **Point-To-Point Link — Packet Switching Characteristics**

<b>Number Of Circuits</b>	The number of identical circuits available to carry packet-switched traffic.
<b>Bandwidth From Node 1</b>	The speed of each circuit in kilobits per second from node 1 to node 2.
<b>Bandwidth From Node 2</b>	The speed of the link in kilobits per second from node 2 to node 1.
<b>Propagation Delay</b>	The physical transmission time across a link. This is generally only significant for satellite (270-300ms) or long distance cable connections (150-200ms).
<b>Session Limit</b>	The maximum number of sessions that can be concurrently routed across the link. If a session setup packet attempts to route across a link which is already at its session limit, the routing attempt fails. The alternate routes (if any) are then tested. If there is still no route the session setup packet is blocked.

**Framing Characteristics**

The framing characteristics for the link. Packets are broken down into transmission frames which are then modeled as being transmitted across the link. (see *Link: Framing* )

## 4. Modeling Constructs

### 4.46 Link: Polling

**Purpose:**

Polling links are typically used to model multidrop telephone lines that use a centralized controller to poll other devices on the line.

**Creating:**

To create a Polling link pick the appropriate link tool from the palette and drag a link onto the background. Or use the create menu option to create a Polling link. On the dialog box accessed via double clicking on the icon you can indicate in the type box to use Polling operation. Once the link has been created a parameter set must be applied to it via a selection in the Parameter pulldown box.

(See *Parameter Sets*)

**Connectivity:**

All types of nodes may connect to a Polling link. As many connections as desired may be made.

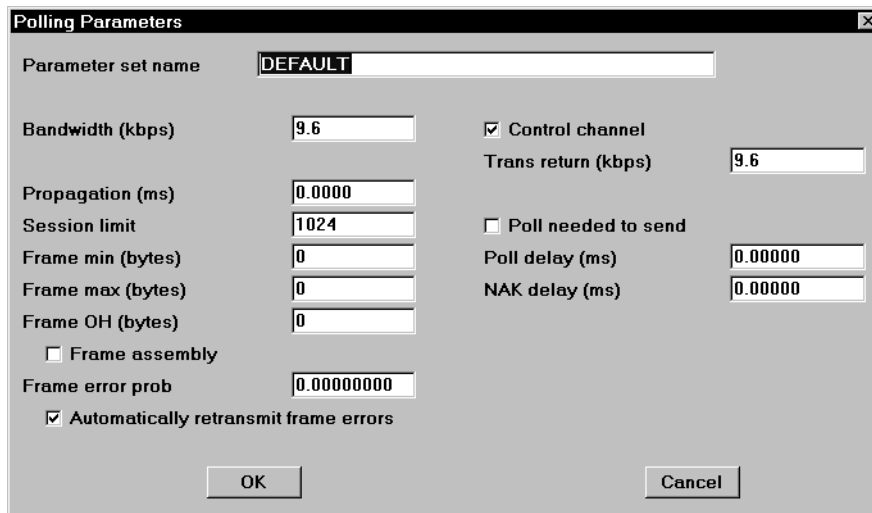
One of the nodes connected to a polling link must be designated the polling node.

Circuit-switched traffic cannot be routed over a Polling link.

**Editing:**

Double click on the icon to bring up the link dialog box, or highlight the icon and pick the Edit/Detail menu option. The name of the link, its icon, and its failure characteristics may then be defined.

Also a parameter set must be applied to the link. The parameter set may be copied from the system library, or created locally within the model. The values found in the parameter set describe the performance characteristics of the link.  
(See *Parameter Sets*)



The image shows a dialog box titled "Polling Parameters" with a close button (X) in the top right corner. The dialog contains several input fields and checkboxes. The "Parameter set name" field is set to "DEFAULT". The "Bandwidth (kbps)" field is set to "9.6". The "Propagation (ms)" field is set to "0.0000". The "Session limit" field is set to "1024". The "Frame min (bytes)" field is set to "0". The "Frame max (bytes)" field is set to "0". The "Frame OH (bytes)" field is set to "0". The "Frame assembly" checkbox is unchecked. The "Frame error prob" field is set to "0.00000000". The "Automatically retransmit frame errors" checkbox is checked. The "Control channel" checkbox is checked. The "Trans return (kbps)" field is set to "9.6". The "Poll needed to send" checkbox is unchecked. The "Poll delay (ms)" field is set to "0.00000". The "NAK delay (ms)" field is set to "0.00000". At the bottom of the dialog are "OK" and "Cancel" buttons.

**Polling Link Parameters Dialog Box**

**Execution:**

A polling node sends, in round robin order, a 'poll' to each node connected to the link.

If a node has packets ready to transmit, it may transmit all of these packets once



the poll has been received. Once the packets have been transmitted then the poll is available to the controller to poll the next node.

Packets will be transmitted in turn from the head of the buffer. For each packet, the link framing characteristics are inspected and the number of frames required for packet transmission calculated. This will result in 0 or more full frames and 1 partial frame (unless the packet size is exactly a whole number of frames). The transmission time for the first frame will be calculated ( $\text{Frame Size}/\text{Circuit Transmission Speed}$ ). If a frame error probability is specified then a statistical pick is made to determine whether the frame would arrive in error. If so, another transmission time is added and another statistical pick made. This process is repeated until a transmission time for the frame is arrived at which delivers the frame without error. The link is then made busy for the calculated transmission time, after which it is made idle. The link is then available for the next frame transmission, but the frame that has just been transmitted must then incur the propagation delay. After the propagation delay, the frame is then at the receiving node. When all frames have arrived at the receiving node, the original packet is reassembled and placed in the input buffer at the receiving node.

If a node has no packets to transmit, it returns a negative acknowledgment (NAK) to the polling node, and the polling node then polls the next node on the link.

A polled node can only transmit on receipt of the poll. The polling node and the polled nodes may transmit over the same channel (half duplex operation), or the polling node may transmit to the polled nodes over a separate control channel (full duplex operation).

The poll itself is not explicitly modeled as data transmitted over the link. This is because the model would then consume much CPU time representing poll transfers, even when the link is otherwise idle. In COMNET III, the poll is represented using a time calculation. When a node has packets ready to transmit it inspects where the poll is currently located and the time it would have left that location, and makes a time calculation for how long it will take to reach itself based on the poll and NAK times. After this length of time the node is then able to transmit its data. Consequently, the overhead of poll representation and management only come into play when there is data waiting to be transmitted over the link.

While a node is transmitting, two or more other nodes may place data in their port output buffers ready for transmission. When the transmitting node completes its transmission the poll is passed in connection order to the nearest node, even if the packets at the other node have been waiting longer. If some intermediate node readies a packet during this poll passing delay the poll will still go to the initially selected node—it will not be interrupted en route to that node. The connection order is the order in which connections were made when the link was defined.

#### ***Reporting:***

##### Link Delays & Utilization Report

Link delays are also inclusive on the Message, Session, and Packet delay reports. Buffering on the link is reported in the Buffer reports.

#### 4. Modeling Constructs

##### ***Fields:***

<b>Name</b>	An alphanumeric field to identify the link. The name must be unique in the backbone or subnetwork.
<b>Icon</b>	The name of the icon used to represent the link.
<b>Control Node</b>	<p>The name of the node which is the control node for the Polling link. A Polling link must have a control node which will act as the poll controller. It is this node which sends the poll to the other nodes on the link and controls the synchronization of transmission.</p> <p>The control node may have an independent, contention free channel to communicate with the other nodes on the link. If such a channel is required, it is specified as part of the Parameter Set for the Polling link.</p>
<b>Link State</b>	Indicates whether the link is in a failed state or not. (see <i>Link</i> )
<b>Statistics</b>	Various statistics may be collected and graphically displayed for Polling Link utilization. These include Contention Channel and Control Channel utilization. (see <i>Statistics: Link</i> )
<b><i>Parameter Set Fields:</i></b>	
<b>Bandwidth</b>	The transmission speed of the Polling link in kilobits per second.
<b>Control Channel</b>	Check box to indicate whether there is a separate, contention free channel for the control node to transmit to the other stations connected to the link. If no control channel is indicated then the transmissions between nodes and the poll controller are half duplex.
<b>Transmission Return Rate</b>	If a control channel is in use, the transmission speed on the control channel.
<b>Poll Needed To Send</b>	<p>If the control node needs to send to one of the other nodes, this flag indicates whether the control node requires the poll before transmission can start, or whether the control node may transmit irrespective of where the poll is.</p> <p>If a control channel is specified then it may be that the control node does not require the poll in order to transmit to other nodes since it has a contention free channel to them, and the control node is the only node which can transmit data over this channel. Turning the “Poll Needed To Send” flag on in this circumstance results in half duplex operation even though there is a full duplex connection between any node and the control node. This is the case for lines that use the BSC (Binary Synchronous Communication) protocol.</p>
<b>Poll Delay</b>	The time it takes for a poll to be sent from the control node to any of the other nodes connected to the link.
<b>NAK Delay</b>	The time it takes a node to send a negative acknowledgment to the control node.

**Propagation Delay**

The time to transmit a signal across a link.

**Session Limit**

In connection-oriented routing, this is the maximum number of sessions that can be concurrently routed across the link. If a session setup packet attempts to route across a link which is already at its session limit, the routing attempt fails. The alternate routes (if any) are then tested. If there is still no route the session setup packet is blocked.

**Framing Characteristics**

The framing characteristics for the link. Packets are broken down into transmission frames which are then modeled as being transmitted across the link.  
*(see Link: Framing)*

## 4. Modeling Constructs

### 4.47 Link: Port

#### *Purpose*

A communications link in COMNET III is represented by an icon on the layout screen. The link has to be connected to the nodes which it supports communication between. This is accomplished by drawing an arc between the link icon and the respective node icons. There is one arc to each node icon. The arc may be thought of as connecting to an i/o port on the node.

There are then input and output buffering parameters which are unique to the port represented by the link/node connection.

When packets/frames are transmitted over a link to a node they are sent from the port output buffer of the transmitting node and received into the port input buffer of the receiving node.

Arcs also hold penalty routing tables for routing algorithms. The arc's penalty tables are applied for traffic entering the link at this arc. *See Penalty Tables.*

#### *Creating:*

Two connection tools are provided to draw arcs. The first allows diagonal lines to be used for the arcs, whereas the other restricts arcs to a combination of horizontal and vertical lines. From a functional point of view they are identical. To create an arc, pick either of the tools and then click on the link followed by the node to indicate the connectivity. Intermediate clicks on the background are taken as corner or hinge points.

With an arc highlighted its buffering parameters may be set via the Edit/Detail menu option, or by double clicking on the arc.

#### *Connectivity:*

One arc may be drawn between a link and a node. Depending on the link type, there may be several nodes connected to a link. A point-to-point link has two connections exactly.

In the backbone network an arc is drawn between a link and the access point of a subnetwork. The access point should be regarded as an extension of the icon of the node in the subnetwork to which it is internally connected. From COMNET III's point of view, the connection is between the backbone link and the subnetwork node. In the subnetwork there is also an arc drawn from the access point to its associated node.

#### *Editing:*

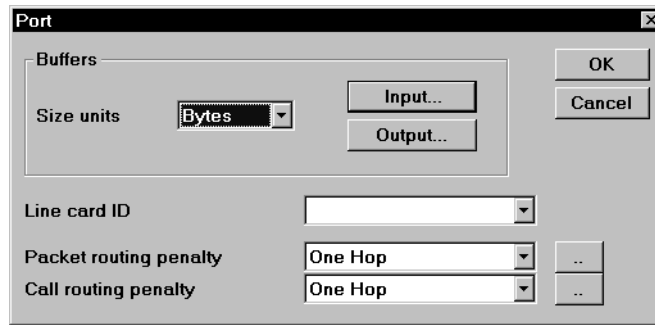
To reconnect an arc it must be deleted and redrawn. To edit the buffer details, highlight the arc and pick Edit/Detail, or double click on the arc.

#### *Execution:*

N/A

#### *Reporting:*

N/A



Port Dialog Box

**Fields:****Input Buffer Size**

The size of the port input buffer on the node associated with the link (the node's port is represented by the arc). If the buffer fills up during the simulation a blocked packet event will occur, which results in retransmission of the packet from its origin (depending on the retry characteristics of the Transport Protocol).

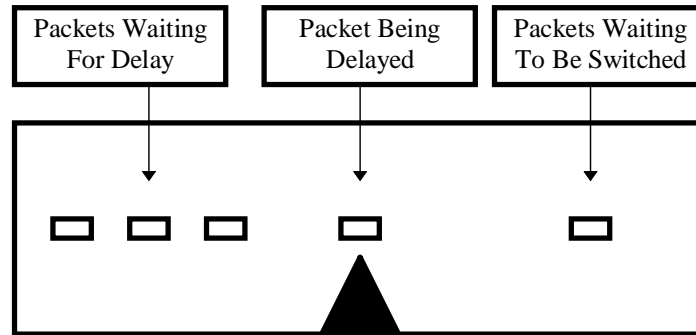
In addition to the input buffer size for the link, a node has a total buffer capacity field which is set on the node parameter screen. The node buffer limit is the total amount of buffering that can be used at any particular instant on the node. It may be that a particular link has capacity in its associated input buffer to receive the packet, but doing so would cause the overall node buffer limit (*see Node Buffers*) to be exceeded. In this case, the packet would also block.

**Input Buffer Delay**

The processing time required at the input buffer. This time is the default amount of time the packet is required to be processed by the port buffer processor. This default time may be overridden by port processing requirements for specific transport protocol IDs. *See Buffer Processing and C&C Nodes.*

When all the frames of the packet have been received the packet is reassembled and placed into the input buffer (assuming there is space available). The buffer status is then updated to reflect this buffer use. The buffer is then modeled as having a single processor that works on packets for the input buffer delay. A packet is not available for removal from the buffer until this time delay has been incurred. If packets are arriving faster than the delay resource can delay them then they are queued in the buffer until the delay resource can process them.

#### 4. Modeling Constructs



**Buffer Diagram**

#### **Output Buffer Size**

The size of the output buffer on the node associated with the link (the node's port is represented by the arc). If the buffer goes full in the simulation a blocked packet event will occur, which results in retransmission of the packet from its origin (depending on the retry characteristics of the Transport Protocol).

In addition to the output buffer size for the link, a node has a total buffer capacity field which is set on the node parameter screen. The node buffer limit is the total amount of buffering that can be used at any particular instant on the node. It may be that a particular link has capacity in its associated output buffer to receive the packet, but doing so would cause the overall node buffer limit to be exceeded. In this case, the packet would also block.

#### **Output Buffer Delay**

The processing time required at the output buffer. This time is in addition to any transmission or propagation delays on the link, packet processing times on the node, or internal switching times on the node.

Once the packet has been placed in the output buffer the buffer status is then updated to reflect this buffer use. However, the packet is not available for transmission until it has been delayed by the output buffer delay (and is at the head of the buffer). The buffer is modeled as having a single processor that works on packets for the output buffer delay. A packet is not available for removal from the buffer until this time delay has been incurred. If packets are arriving faster than the delay resource can delay them then they are queued in the buffer until the delay resource can process them.

#### **Line Card ID**

This attribute is only significant when the port connects to a router node. Otherwise it is ignored.

For a router it is common to have line interface cards that connect the router to one or more links. If the routing decision takes a packet in and out on the same line card then the switching time is comparatively fast. If the routing decision requires the packet to be moved across the router between line cards then the switching time will be longer. In COMNET III, a routing decision that results in different line card id's being used will force transmission over the internal bus of the router (which has its own speed and therefore additional delay).

On the link arc, you may specify the line card id. Arcs that have the same line

non-blank card id will be treated as connecting to one card and so traffic routing between them will not see bus delays. Arcs that have different or blank line card id's will be treated as connecting to different line cards and so traffic routing between them will have to also incur a bus delay.

**Call Routing Penalty Table**

When penalty based routing is specified for the backbone or subnetwork in which the link is defined, the link must have an associated penalty table so that its cost can be looked up. The table name to use is specified on the arc detail—the table itself is constructed on the **Define/Penalty Table** menu option.

The table is defined from the point of view of the node from which routing will be attempted. A point-to-point link could therefore have a penalty table defined for routing in one direction, and a separate penalty table for routing in the other direction. This allows the user to prohibit traffic flow in a particular direction on a link.

*(see Penalty Table: Call Routing)*

**Packet Routing Penalty Table**

The same as for the call routing penalty table, except this defines the table to use for packet routing. For a multiaccess link such as CSMA/CD, each node could be use a different penalty table for the link.

*(see Penalty Table: Packet Routing)*

## 4. Modeling Constructs

### 4.48 Link: Priority Token Ring and Token Ring Enhancements

*Purpose:*

Release 1.3 improves the token passing model for better performance, more flexibility, and better monitoring. In addition new link types derived from the token ring model provide better models for priority token-rings (including 100VG-AnyLAN), and FDDI.

The token-passing model has a new option for a holding limit based on the number of frames instead of a holding time, and this may be particularly useful when the node may hold a token for exactly one frame.

New statistics monitors and a token-ring report are available to measure the actual token rotation times, and the number of active nodes waiting for the token and how long these nodes remain actively accessing the link. These monitors and reports are available for all the token-passing link types (token passing, priority token ring, FDDI, priority FDDI).

The priority token-ring model provides parameter sets for IEEE 802.4 and 802.5 token rings, and 2 new parameter sets for 100VG-AnyLAN: 1 set for ethernet ports and 1 set for token ring ports.



## 4.49 Link: Token Passing

- Purpose:** Token-Passing links are used to model 802.4, 802.5 and FDDI operation.
- Creating:** To create a Token-Passing link pick the appropriate link tool from the palette and drag a link onto the background. Or use the create menu option to create a Token-Passing link. On the dialog box accessed via double clicking on the icon you can indicate in the type box to use Token-Passing operation. Once the link has been created a parameter set must be applied to it via a selection in the Parameter pull-down box.  
(See *Parameter Sets*)
- Connectivity:** All types of node may connect to a Token-Passing link. As many connections as desired may be made.  
  
Circuit-switched traffic cannot be routed over a Token-Passing link.
- Editing:** Double click on the icon to bring up the link dialog box, or highlight the icon and pick the *Edit/Detail* menu option. The name of the link, its icon, and its failure characteristics may then be defined.  
  
Also a parameter set must be applied to the link. The parameter set may be copied from the system library, or created locally within the model. The values found in the parameter set describe the performance characteristics of the link.  
(See *Parameter Sets*)

Parameter	Value
Parameter set name	DEFAULT
Bandwidth (kbps)	16000
Token passing delay (ms)	0.0000
Token holding limit (ms)	0.0000
Propagation (ms)	0.0000
Session limit	1024
Frame min (bytes)	0
Frame max (bytes)	0
Frame overhead (bytes)	0
Frame assembly	<input type="checkbox"/>
Frame error prob	0.00000000
Automatically retransmit frame errors	<input checked="" type="checkbox"/>

**Token Passing Link Parameters Dialog Box**

- Execution:** A Token-Passing link sends, in round robin order, a 'token' to each node connected to the link.

When the token is received the node may transmit packets up to the token hold-

#### 4. Modeling Constructs

ing time. Once the packets have been transmitted, or the token holding time expires, the token is passed to the next node. If there are no packets to transmit the token is passed on immediately.

Packets will be transmitted in turn from the head of the buffer. For each packet the link framing characteristics are inspected and the number of frames required for packet transmission calculated. This will result in 0 or more full frames and 1 partial frame (unless the packet size is exactly a whole number of frames). The transmission time for the first frame will be calculated ( $\text{Frame Size} / \text{Circuit Transmission Speed}$ ). If a frame error probability is specified then a statistical pick is made to determine whether the frame would arrive in error. If so another transmission time is added and another statistical pick made. This process is repeated until a transmission time for the frame is arrived at which delivers the frame without error. The link is then made busy for the calculated transmission time, after which it is made idle. The link is then available for the next frame transmission, but the frame that has just been transmitted must then incur the propagation delay. After the propagation delay the frame is then at the receiving node. When all frames have arrived at the receiving node the original packet is reassembled and placed in the input buffer at the receiving node.

If the token holding time expires halfway through a frame transmission, the frame transmission is completed and then the token is passed on.

The token itself is not explicitly modeled as data transmitted over the link. This is because the model would then consume much CPU time representing token transfers, even when the link is otherwise idle. In COMNET III the token is represented using a time calculation. When a node has packets ready to transmit it inspects where the token is currently located and the time when it would have left that location, and makes a time calculation for how long it will take to reach itself based on the token passing times. After this length of time the node is then able to transmit its data. Consequently, the overhead of token representation and management only come into play when there is data waiting to be transmitted over the link.

While a node is transmitting, two or more other nodes may place data in their port output buffers ready for transmission. When the transmitting node completes its transmission the token is passed in connection order to the nearest node, even if the packets at the other node have been waiting longer. If some intermediate node readies a packet during this token passing delay the token will still go to the initially selected node—it will not be interrupted en route to that node. The connection order is the order in which connections were made when the link was defined.

#### **Reporting:**

Link Delays & Utilization Report

Link delays are also inclusive on the Message, Session and Packet delay reports. Buffering on the link is reported in the Buffer reports.

#### **Fields:**

##### **Name**

An alphanumeric field to identify the link. The name must be unique in the backbone or subnetwork.

<b>Icon</b>	The name of the icon used to represent the link.
<b>Link State</b>	Indicates whether the link is in a failed state or not. (see <i>Linksee Node</i> )
<b>Statistics</b>	Various statistics may be collected and graphically displayed for Token Passing Link utilization. (see <i>Statistics: Link</i> )
<b><i>Parameter Set Fields:</i></b>	
<b>Bandwidth</b>	The transmission speed of the Token-Passing link in kilobits per second.
<b>Token Passing Delay</b>	The time it takes for a token to be sent from one node to the next node.
<b>Token Holding Limit</b>	The time limit for a node to hold the token. A limit of 0 implies keeping the token until all the packets have been transmitted.
<b>Propagation Delay</b>	The time to transmit a signal across a link.
<b>Session Limit</b>	For connection-oriented routing, this is the maximum number of sessions that can be concurrently routed across the link. If a session setup packet attempts to route across a link which is already at its session limit, the routing attempt fails. The alternate routes (if any) are then tested. If there is still no route the session setup packet is blocked.
<b>Framing Characteristics</b>	The framing characteristics for the link. Packets are broken down into transmission frames which are then modeled as being transmitted across the link. (see <i>Link: Framing</i> )

## 4. Modeling Constructs

### 4.50 Message Source: Interarrival Time

**Purpose:**

When multiple messages are created the time between successive instances of messages must be defined. The specified time is from the start of one message to the start of the next message.

Any of the allowed COMNET III distributions (including Table Distributions and User Defined Distributions) may be used for specifying interarrival time. (see *Distributions*)

The most common distribution to use for interarrival times is the exponential distribution. The parameters entered for the exponential are the mean value and the random stream number to use.

Telecommunications traffic is often described as a Poisson process. This generally means that the number of messages in successive time intervals have been observed and the distribution of the number of observations in an interval is Poisson distributed.

In COMNET III the number of messages per unit of time is not entered. Rather, the interarrival time between messages is required. It may be proved mathematically that, if the number of messages per unit time interval is Poisson-distributed, then the interarrival time between successive messages is exponentially distributed.

**Creating:**

To use a particular distribution use the pull down list next to the interarrival-time distribution box on the dialog box for the particular message generator.

**Message Source Dialog Box**

**Applicability:**

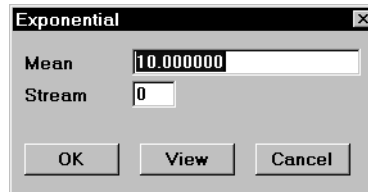
Message interarrival times are required as part of session commands, session

sources, and message sources.

**Editing:**

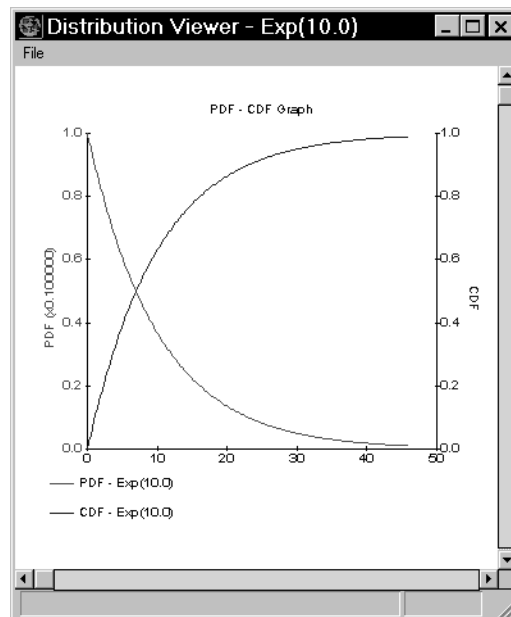
The parameters of the distribution can be directly entered in the text box on the message generator dialog box.

Alternatively, the “double dot” button next to the pull down arrow may be selected after which a dialog box appears which has an individual value box for each field in the distribution.



**Exponential Distribution Parameters**

At this point the distribution may also be viewed as a visual check on its appearance.



**Exponential Distribution Curve**

**Execution:**

An interarrival time is the time from the start of one message to the start of the next message. This does not mean that the first message has to complete before the second message starts. If the message delivery time turns out to be long, or an earlier message blocks at some point, it may well be expected that multiple messages from the same traffic generator overlap.

Overlapping messages are intentional where a message generator is modelling

#### 4. Modeling Constructs

the creation of messages from a number of users. Maybe the node at which the generation occurs represents a pad, concentrator or terminal server and the aggregate rate of message generation from it is required rather than modelling each individual user.

If no specific start time is entered for the message generator, a time  $T$  will be picked from the interarrival-time distribution, and the first arrival will be chosen from the uniform distribution between time 0 and  $T$ . When the first message is created then another statistical pick is made from the given distribution to determine when the next message should be created. This process continues until the end of the simulation or the message generator stop time (if one is entered).

**Reporting:**

N/A

**Fields:**

**Start Time**

The time in seconds from the start of simulation (time = 0) when the first message should be scheduled. This may be a constant, or a statistical pick.

If nothing is entered (Start Time = None), a time  $T$  is drawn from the interarrival-time distribution and the first arrival time is then drawn uniformly from  $[0, T]$ .

If a constant of 0 is entered, the first message will go at time 0.

If the interarrival time is None and the Stop Time is None and the Start Time has some constant value (5 say), then only 1 message will be created at the indicated time.

**Interarrival Time**

The time from the start of one message to the start of the next.

**Stop Time**

The time in seconds from the start of simulation (time = 0) when the last message should be scheduled. This may be a constant, or a statistical pick.

If nothing is entered (Stop Time = None), the end of simulation is the stop time.

If a Stop Time is entered a message will be scheduled at this time. For example, if the first message was scheduled at 0.5 seconds, the Interarrival Time was 10 seconds and the Stop Time 11 seconds, the next message would be created at 10.5 seconds and the last one at 11 seconds.

## 4.51 Message Source: Packetizing Delay

### *Purpose:*

Nodes are modeled as having a node processor which, at packet creation, is made busy for the packetizing delay. In other words, the packetizing delay is the amount of processing time it takes to make a packet at the originating node.

The packetizing delay could be used to model transaction processing time at a host. Messages which represent transactions may be sent to a node which represents a host. Upon receipt of the message the host may be required to echo data back to the sender (e.g., a credit card authorization network) via a response source. If the response message is a single packet, a simple way to model the processing delay on the host would be to specify a packetizing delay. If the host could process 100 transactions per second (say) this would imply a packetizing delay of 10 milliseconds. As response messages are made, each has to incur the 10 millisecond processing delay on the host processor, which is then effectively a single server, single queue resource. If the host were then upgraded to 200 transactions per second then the only parameter to change would be the packetizing delay (to 5 milliseconds). Obviously a more detailed model could be made by using read, write, processing and transport commands from within an application running on the host—but a simple response source with a packetizing delay may suffice as a first pass model.

The packetizing delay might also be used to model transport layer processing overhead.

### *Creating*

The packetizing delay is specified on the traffic generators that are attached to the originating node, or as part of traffic generating commands (session setup, answer message, transport command). This allows the packet creation time to vary by application and type of traffics.

### *Connectivity:*

Defined on each message generator.

### *Editing:*

Edited via the message generator dialog box.

### *Execution:*

Packets are created as a result of a transport command, session setup command, or response command being picked for execution on a node. Message, session, and response sources result in internally generated applications with a respective command to perform this function. When the node schedules the application to execute its command, the transport protocol is queried to establish that any outstanding ACKs have been received. If these requirements are satisfied the packet is created and the node processor made busy by the packetizing delay. Also a check is made that buffer space is available and that a route can be found.

The sequence of events that occurs is as follows:

1. Create message
2. Incur packetizing delay
3. Create packet
4. Incur the packet processing delay
5. Route the packet.

Session setup packets, session connect packets and ACK packets do not incur the packetizing delay. The packets which represent the message data do incur the

#### 4. Modeling Constructs

packetizing delay.

***Reporting:***

Packetizing delay counts towards Node Utilization.  
It is also part of the respective Message Delay and End-To-End Packet Delay.

***Fields:***

**Packetizing Delay**

Use any of the standard COMNET III distributions.  
(see *Distributions*)



## 4.52 Message Source: Priority

**Purpose:** The priority field is used to assign a priority to packets as they are created. Packet priority is used to order packets in input and output port buffers on nodes, with highest priority packets being placed at the head of queue of packets in the buffer. Packets of equal priority are treated on a FIFO basis.

The Priority field is not used to imply priority between multiple traffic generators or applications on the same node. Multiply scheduled generators are executed on a round-robin basis.

(See *Node Scheduling*)

Also the Priority field is not used to arbitrate between packets waiting at different nodes to access a link. It is up to the link protocol to choose which packet to transmit from those packets at the head of the output port buffers of the transmitting nodes. For instance, in token passing the next packet in connection order would be serviced—irrespective of the priority of packets on other buffers. See *Link: Token Passing*

**Creating:** Priority is defined on the dialog box which describes the message or transport command which is going to create the packet.

**Connectivity:** N/A

**Editing:** Via the dialog box.

**Execution:** When a packet is created it is assigned a priority based on the priority entered for the message generator dialog box. After creation, the packet incurs node processing delays followed by a routing decision which determines which output port buffer to route through. The packet is then placed in the selected output port buffer.

The buffer may have a buffer processing time. In this case the buffer is modeled as having a single delay resource which is used to delay the packets as they are undergoing processing in the buffer. The packet is inserted into the buffer behind this delay resource. If there are multiple packets behind the resource they are ordered in decreasing priority order, then by FIFO. Consequently the highest priority packet that has waited the longest is allocated the delay resource first and incurs the buffer processing delay first. Other low priority packets must wait for the higher priority packets to be delayed in the buffer before the low priority packets may incur their delay. Once packets have incurred the buffer processing delay they are then waiting for transmission across the connected link. Packets waiting for transmission are again ordered in decreasing priority followed by FIFO order.

At some point in time the link protocol will determine that the packet at the head of the port output buffer queue is to be transmitted. The link protocol transmits the packet frame-by-frame until all of the frames have arrived at the next node. The packet is then reassembled and placed in the input port buffer on the receiving node. Again the input port buffer is ordered in decreasing priority plus FIFO order, and has an associated delay resource to model processing delays into the buffer.

#### 4. Modeling Constructs

A packet with some origin and destination is routed from node to node by the routing algorithm in force for the network. At each intermediate node it is received into the input port buffer and switched into an output port buffer. The manner in which the packet is switched and the associated delay depends on the type of node and its parameter settings. Each buffer exhibits the same logic with respect to packet priority.

***Reporting:***

N/A

***Fields:***

**Priority**

An integer value between 1 and 99 that is used to set the priority of packets generated by the message source. Larger values have higher priority.

## 4.53 Message Source: Size Calculation

**Purpose:** The message size field is used to assign a total size in bytes or packets to each message as it is generated. If a message source is being used to model electronic mail, file transfers, database inquiries, etc., then the size of the data transfer needs to be described to the model. Message sizes are not necessarily constants—for instance, not all file transfers are of the same size. Consequently COMNET III provides for the message size to be entered as a statistical distribution.

Also, it may be that a message size needs to be related to the size of an inbound message, or the size of data which has been read from disk. Such provision is made within COMNET III.

If the message is sized in bytes, the transport protocol settings will be inspected to determine how many full packets represent the message (with possibly one, smaller overflow packet).

If the message size is given as N packets, COMNET III creates messages of size N times the packet size, so all COMNET III messages internally use bytes as the size unit, once the message has been created. When these messages are packetized, there will be no overflow packets.

**Creating:** Message size is defined on the dialog box which describes the message or transport command.

**Connectivity:** Messages are generated by Message Sources, Response Sources, Session Sources, Transport Commands, Setup Commands and Answer Commands. Wherever a message may be generated, a message size definition may be entered.

**Editing:** Via the appropriate message generator dialog box.

Any of the standard COMNET III distributions may be used.  
(see *Distributions*)

**Execution:** When a message is created a statistical pick is made from the message size distribution to determine the total size of the message. This may be in either bytes or full size packets. If the message size is in bytes a calculation of how many packets are required is made, based upon the transport protocol settings. Assuming a statistical distribution is being used (rather than a constant), then as the simulation proceeds and different messages are generated, each message will have its own, individual size as returned by the statistical pick.

The originating node then creates the required number of packets one after the other taking note of node scheduling requirements, flow control settings, routing decisions, etc., until all of the packets have been generated.

**Reporting:** N/A

### **Fields:**

**Message Size** Three choices are available for setting the message size. These include:

**Probability Distribution** (see *Distributions*)

#### 4. Modeling Constructs

Any of the allowed COMNET III distributions.

##### **Based On Message Size**

If the application or message source which wants to create the message was itself scheduled by received message, the size of the scheduling message can be used to calculate the size of the reply. This is done using a linear calculation where a multiplier and an offset can be specified. The size calculation is then of the form:

$$\text{Multiplier} * \text{Message Size} + \text{Offset}$$

If the multiplier is 1 and the offset is 0 then a message of the same size will be returned.

##### **Based On File Size**

If the application which wants to create the message has read a file, the size of the first read can be used to calculate the size of the message. This is done using a linear calculation where a multiplier and an offset can be specified. The size calculation is then of the form:

$$\text{Multiplier} * \text{Number of Bytes Read} + \text{Offset}$$

If the multiplier is 1 and the offset is 0 then a message of the same size as the number of bytes read from file will be created.

## 4.54 Message Source: Size Units

<i>Purpose:</i>	Messages may be sized in bytes or packets. If they are sized in packets, then the message size calculation specifies the number of full size packets, where the packet size is derived from the transport protocol settings. Note that the Message Linear distributions always use the received message size in bytes.
<i>Creating:</i>	Message Size Unit is defined on the dialog box which describes the message or transport command dialog box.
<i>Connectivity:</i>	Messages are generated by Message Sources, Response Sources, Session Sources, Transport Commands, Setup Commands and Answer Commands. Wherever a message may be generated a message size unit definition may be entered.
<i>Editing:</i>	Via the appropriate message generator dialog box.
<i>Reporting:</i>	N/A
<b>Fields:</b>	
<b>Size Unit</b>	Only the values “Bytes” or “Packets” may be selected.

## 4. Modeling Constructs

### 4.55 Message Source: Message Text

***Purpose:***

When a message is sent, there is an arbitrary text label associated with the message. This message text is particularly used to trigger sources based on received message scheduling for application, message, response and session sources. Received Message Scheduling means that when a message with the required message text arrives at the destination node then the scheduling requirement for that application at the destination node has been met and may be started.

For instance, a model of a client server architecture may be built. Each client has a message source which sends a query message to the server with the message text “Database Query.” On the server there may be an application which is scheduled by received message and is looking for the message text “Database Query” to be received. When a message with this text is received a new instance of the server’s application may be started.

Because the message text is separate from the message name, a number of different message sources may send the same message text and thus trigger the same source at a destination, despite differences of the message itself (in terms of size, source, protocol, etc.).

Message text can be useful for measuring message delays: essentially, the message delays reported in the output reports measure the “lifetime” of the message text. If a message uses the original message text, the delay for that message is measured from the creation time of the incoming message text. This is a useful mechanism for getting round-trip delays.

***Creating:***

Message Text is defined on the dialog box which describes the message or transport command.

***Connectivity:***

Messages are generated by Message Sources, Response Sources, Session Sources, Transport Commands, Setup Commands and Answer Commands. Wherever a message may be generated, a message text may be entered.

***Editing:***

Via the appropriate message generator dialog box.

***Execution:***

When a message is created, its message text is set based upon the message text settings. This is either a copy of the message name, the text of the original message, or specific text entered by the user. This text, or label, is then associated with the message as it moves to its destination.

When the message arrives at the destination its text may be used to schedule some activity at the destination, such as a message source or a response source or an application. Scheduling can be made to depend on more than one incoming message: through a received message list identifying

- Required number of a message text (all must be received) to trigger.
- list of different message texts (all must be received)

Each message text in the received message list may use the wildcard character “\*” to accept classes of message texts. For example, the following are examples using the wildcard character “\*”:

\*  
 \*Request  
 Request\*  
 \*Request\*

The first case, where the wildcard character is by itself, will match everything. The second case, where the wildcard character appears first, will match any received message that ends with “Request.” Similarly, the third case will match any received message that begins with “Request.” Finally, with the wildcard character appearing at both the beginning and end of the message, any received message that contains “Request” anywhere in the text will be matched.

If the message text is not required by any source at the destination the message and its associated text are discarded.

If the text is specified as a requirement by some source at the destination then the message text is placed in a received message list at the destination and remains there until such time as it (or rather its message text) is used to satisfy a scheduling requirement. A particular message instance will only satisfy one scheduling requirement at the destination, after which the message is considered used and so discarded.

*(See Received Message Scheduling)*

**Reporting:**

N/A

**Fields:**

**Message Text**

From the sender of the message point of view, the text of the message must be set. The options for setting the text are:

**Use Original Message**

Use the text of the message which scheduled the application, response, etc., which is creating the new message.

**Copy Message Name (Default)**

Use the name of the message as the message text.

**Set Message Text**

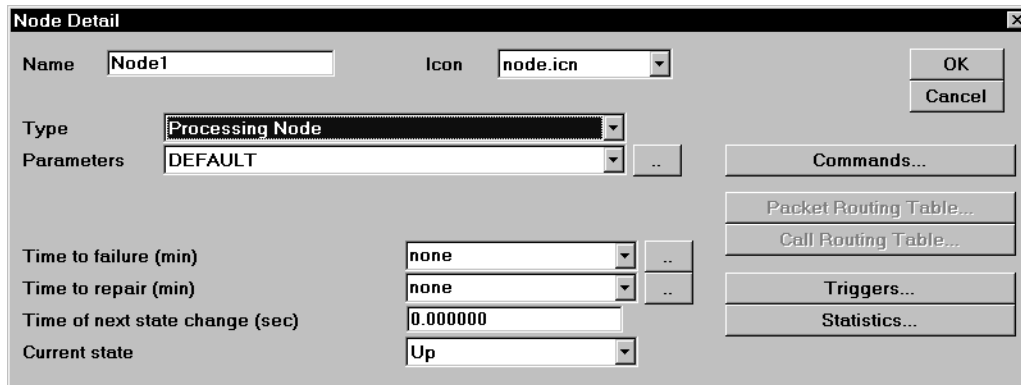
Explicitly set the message text with a label typed by the user.

## 4. Modeling Constructs

### 4.56 Node

**Purpose:**

A node in COMNET III models switches, hubs, routers, end systems, pads and general network components. Specific types of nodes are ATM, Computer and Communications (C&C), Computer Group, and Routers.



**Node Detail Dialog Box**

**Creating**

To create a node, choose one from the palette and drag it onto the background. You can also create one by choosing Create/Nodes... from the menu bar.

**Connectivity:**

For details about which types of links connect to which types of nodes, see the section on the appropriate node.

**Editing:**

To edit a node, either click once on the node to select it and then choose Edit/Detail, or double-click on it. Both methods bring up the Node Detail dialog box.

**Fields:**

**Name:**

The name of the node. The name must be unique in the backbone or subnetwork.

**Icon**

The name of the icon used to represent the node.

**Type**

The type of node: ATM, C&C, Computer Group, Router or one defined by the user.

**Parameters**

Parameters for that type of node. Parameters may include specifications for application processing, packet processing, port processing, circuit switching and disk storage and will depend on the node type.

**Time to failure (min)**

Nodes may be made to fail at a time determined by a probability distribution.

**Time to repair (min)**

The time it takes a node to be repaired can be set by a probability distribution.

**Time of next state change (sec)** A time at which the state of the node will be toggled: if the node is currently up the node will be brought down; if the node is currently down it will be brought back on-line. The time is given in terms of the simulation time. For example,



if the Time of the next state change is 30 seconds, then 30 seconds into each replication the state will be changed.

**Current state**

Whether the node is currently operating (up) or failed (down). This can be toggled during the simulation by double-clicking on the node to bring up the node detail dialog box and editing this field.

**Commands**

Brings up the Command Repertoire dialog box. Here commands can be added or removed from the node's repertoire. For more information about commands, see the section on the specific type of command: Answer Message, Filter Messages, Process Data, Read File, Setup Session, Transport Message, or Write File

## 4. Modeling Constructs

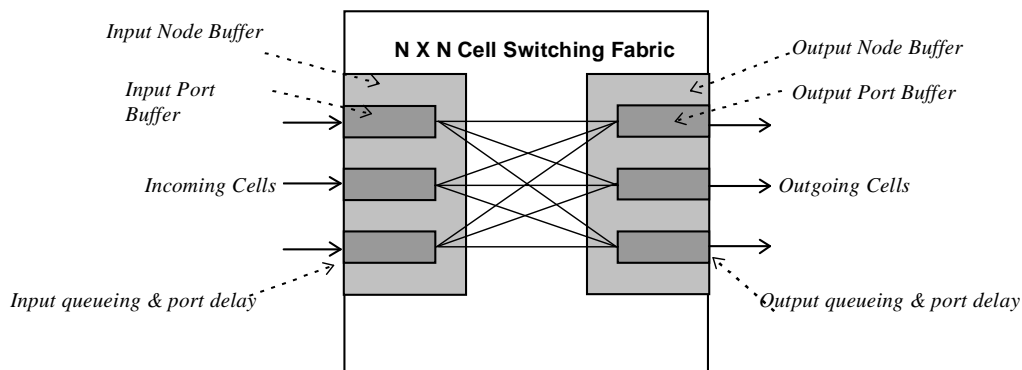
### 4.57 Node: Switch

#### *Purpose:*

A Switch Node is aimed at modeling switches, hubs, or routers which either utilize high-bandwidth internal switching fabric (no contention between in an output buffers) or have insignificant latency delays compared to other delays in the model. The basic premise is that for a Switch Node the internal architecture of the switch is designed to be non-blocking, and that the performance of the switch can be effectively captured via its input and output buffering characteristics.

Consequently, in COMNET III a Switch node has input and output ports which have associated buffer sizes and buffer processing times, but no other internal architecture of the node is modeled. Once a packet has incurred the buffer processing delay at the port input buffer through which it received, it is instantaneously switched (if there is space available) to a port output buffer as determined by the routing algorithm.

To model ATM cell transfer systems, where message generators create packets, the packet which is created should be made equivalent to a cell. For instance, with ATM, cell sizes are 48 bytes payload and 5 bytes overhead. In COMNET III the message generator should have a transport protocol with a maximum packet size of 48 and a packet overhead of 5. The ATM cell and the COMNET III packet are then synonymous.



**ATM Node Diagram**

#### *Creating:*

To create a Switch node pick the appropriate node tool from the palette and drag a node onto the background. Or use the create menu option to create a Switch node. On the dialog box accessed via double clicking on the icon, you can indicate in the type box to use ATM operation if the node you have created is of a different type. Once the node has been created a parameter set must be applied to it via a selection in the Parameter pulldown box.

(See Parameter Sets)

#### *Connectivity:*

All types of link may connect to a Switch node. As many connections as desired may be made. For each link connected to the node there is an input port buffer and an output port buffer. The details of these buffers may be accessed via double clicking on the arc which connects the link to the node.

Circuit-switched traffic may be routed through a Switch node, but may only use

point to point links.

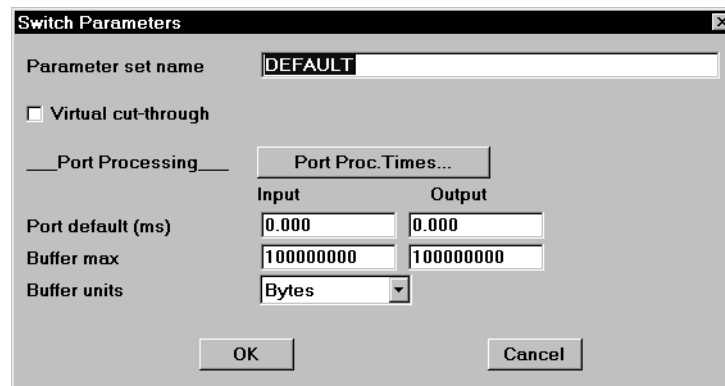
a Switch node cannot be an origin or a destination for traffic. It may only act as a switching point.

**Editing:**

Double click on the icon to bring up the node dialog box, or highlight the icon and pick the **Edit/Detail** menu option. The name of the node, its icon, and its failure characteristics may then be defined.

Also a parameter set must be applied to the node. The parameter set may be chosen from the model's list. The values found in the parameter set describe the performance characteristics of the node.

(See *Parameter Sets*)



**Switch Parameters Dialog Box**

**Execution:**

**Packet Switching**

Normally, a packet arrives at a node when all the frames which constitute the packet have been delivered to the node by the link protocol. The packet is reassembled from its frames and an attempt is made to insert the packet into the input port buffer which services the link. [Optionally, an incoming packet can begin transmission on an outgoing link while bits are still arriving on the incoming link. This option is called Virtual cut-through.]

A test is made to see whether there is sufficient space in the input port buffer to receive the packet. If there is sufficient space a test is made to see whether there is sufficient space in the overall Switch node input buffer limit. If both tests pass then the packet is placed in the port input buffer which services the link. If either test fails then the packet is considered blocked and may be retried from the origin depending on the transport protocol settings.

The packets in the buffer are ordered in decreasing priority plus FIFO order. Once in an input buffer a packet must incur a buffer processing delay before it may be considered for removal from the buffer. The buffer is modeled as owning a single delay resource which is used to incur the packet delay, one packet at a time. Consequently the packets in the input port buffer may have to wait until they are at the head of the queue in order to incur their delay.

## 4. Modeling Constructs

A packet occupies buffer space while it waits its turn to be processed at the port. Once it has finished processing, it continues to occupy input buffer space while it waits its turn to be processed by the node processor. After it is processed by the node, a routing decision is made that determines which output port the packet should use (assuming the packet is not yet at its destination).

The packet is then moved from the input buffer to the selected output buffer. If the input and output ports have the same non-blank line card IDs, then the packet is transferred from the input to the output buffer without traversing the router's bus. If the line card IDs are different, the packet must traverse the bus before entering the selected output buffer. If there is contention for the router's bus (or buses), the packet will continue to reside in the input buffer until it can be transmitted on the bus. (*See Routing Protocol: Packet*)

The attempt to move the packet may fail because the selected port output buffer has insufficient space available. Again, two tests are made against the available port output buffer space and the overall Switch node output buffer space. Both tests must pass for the packet to be moved to the selected output buffer.

If either of the buffer tests fail then the packet is blocked, and remains at the head of the input port buffer. This is in contrast to C&C node and Router node processing, where blocked packets would be dropped and optionally retransmitted. The packet will remain at the head of the port input buffer queue until either space becomes available in the selected output port buffer, or a higher priority packet arrives and so moves in front of the blocked packet. However, other packets waiting behind the blocked packet will have to wait for the blocked packet to be switched before they can be considered for switching. This typifies Head Of Line blocking as seen in many ATM switch implementations.

If there are several packets simultaneously at the head of different port input port buffers they will all be switched instantaneously to their selected output port buffers (assuming no blocking). Instantaneously means at the same point in simulation time. In other words, the Switch node can switch in parallel as many packets as required between input and output buffers. The sequence in which input buffers are served is in connection order.

Once a packet is placed in the port output buffer it must incur the buffer processing delay for the port. Similar to the input port, the packets queue in decreasing priority and FIFO order for the single delay resource, are delayed by the specified time, and then wait in the output buffer in decreasing priority and FIFO order to be transmitted by the downstream link.

### Circuit-Switching

When a circuit-switched call originates, a sequence of routing decisions is typically required, depending on the number of routes composing the end-to-end path and the number of hops in each route. The nodes and links along a possible route are inspected to determine whether the complete route has capacity to carry the call. From the point of view of a Switch node, the node has a maximum possible bandwidth which it can concurrently switch for all circuit-switched calls routed through it. As a call is routed through the Switch node, the amount of remaining bandwidth available for circuit-switched traffic is reduced. When a call clears the used bandwidth is given back.

Consequently, a Switch node is available to carry a particular call if, at the instant the routing decision is being made, the amount of free bandwidth on the Switch

node is greater than (or equal to) the bandwidth requirement for the call (as defined on the routing class dialog box).

Only when the routing decision has checked that a route has sufficient capacity is the bandwidth claimed. However, it should be noted that this is an instantaneous decision.

If the Switch node is a congestion point for the circuit-switched traffic (i.e., it is a point on the route where there is insufficient bandwidth), the routing protocol will first of all test other routes to see if one is available. If no other route is available, the routing protocol may revisit the blocking node and test whether there are any calls routed through the node which may be preempted to release sufficient bandwidth to route the required call. The preemption mechanism is controlled by call priority as set on the call source dialog box and preemptions must also be activated on the call routing protocol dialog box.

(See *Routing Protocol: Call*)

**Reporting:**

The buffer reports indicate buffer congestion on the Switch node. See: *Input Buffer Report By Port*

Input Buffer Report By Node

Output Buffer Report By Port

Output Buffer Report By Node

The Message Delay and Packet Delay reports reflect delays at the ATM node where the Switch node is part of the selected route.

**Fields:**

**Name** An alphanumeric field to identify the node. The name must be unique in the backbone or subnetwork.

**Icon** The name of the icon used to represent the node.

**Node State** Indicates whether the node is in a failed state or not.  
(see *Node*)

**Packet Routing Table** (see *Routing Table: Packets*)

**Call Routing Table** (see *Routing Table: Calls*)

**Parameter Set Fields:**

**Parameter Set Name** The name you wish to use for a specific implementation of switch parameters.

**Virtual cut-through** The virtual cut-through operation allows an incoming packet to begin transmission on an outgoing link while bits are still arriving on the incoming link. When cut-through is enabled for a node, the node must be connected to at least two links, and all connected links must be point-to-point, have the same speed, and be configured so the size of a packet and a frame are the same. When a packet begins transmission to a cut-through node, the header of the packet is received and inspected to determine the next link the packet will take, and if that link is available, the packet immediately begins transmission on the outgoing link, even

#### 4. Modeling Constructs

before the rest of the packet has arrived. Thus, for a packet to go from A to B to C, where B is a cut-through node, the expected delays from A to B and from B to C will overlap in time, reducing the total delay: This pipelining effect occurs because transmission of a packet on an outgoing link can begin before the entire packet has been received on an incoming link. There is no link-level flow control or error checking over links connected to cut-through nodes.

##### **Input Buffer Max**

The maximum allowable of input buffer usage across all input ports connected to the Switch node. When a packet is received, a test is first made to see if there is space in the port buffer, and then a test is made to see whether the total buffer usage limit (as defined here) would be exceeded if the packet were accepted.

The input port buffer size is defined on the arc which connects a particular link to the Switch node.

If there is insufficient space to receive a packet then the packet is blocked and optionally retried depending on the transport protocol settings.

##### **Output Buffer Max**

The maximum allowable output buffer usage across all output ports connected to the Switch node. When a packet is switched to an output port, then a test is first made to see if there is space in the port buffer, and then a test is made to see whether the total buffer usage (as defined here) would be exceeded if the packet were accepted.

The output port buffer size is defined on the arc which connects a particular link to the Switch node.

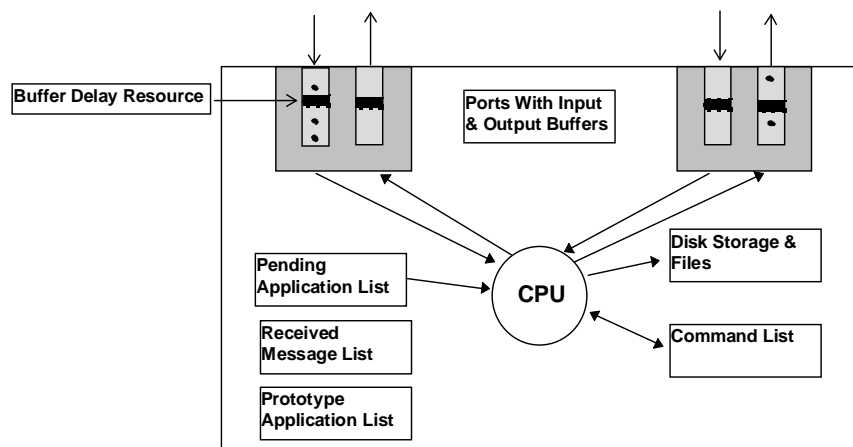
If there is insufficient space to receive a packet into the output port buffer, then the packet is blocked and remains in the input port buffer until space becomes available. This treatment of holding blocked packets is unique to the Switch node.

## 4.58 Node: Processing Node

### *Purpose:*

The “generic” node used to represent end systems, packet switches, pads, and general network components. The C&C node may originate all types of traffic, route both data and circuit-switched calls, and execute applications. Applications are used to represent a program or subroutine running on the modeled device. A specific application requests the node to execute a user defined sequence of read, write, process, transport, setup or answer commands. The C&C node is represented in the model as having:

- An input buffer for each link transmitting packets to it.
- A processor to execute commands and process packets.
- An output buffer for each link to which it can route packets.
- Local disk storage capacity for modelling local read/write commands.
- A command list which details how particular commands are executed.
- A pending application list of currently scheduled applications.
- A prototype application list of available applications at the node.
- A received message list for saving received messages until they are used.
- A list of files that may reside in local disk storage.



### Computer & Communications Node Architecture

#### *Creating:*

To create a C&C node, pick the appropriate node tool from the palette and drag a node onto the background. Or use the create menu option to create a C&C node. On the dialog box accessed via double clicking on the icon, you can indicate in the type box to use C&C operation if the node you have created is of a different type. Once the node has been created, a parameter set must be applied to it via a selection in the Parameter pulldown box.

(See *Parameter Sets*)

## 4. Modeling Constructs

### *Connectivity:*

All types of links may connect to a C&C node. As many connections as desired may be made. For each link connected to the node there is an input port buffer and an output port buffer. The details of these buffers may be accessed via double clicking on the arc which connects the link to the node.

Circuit-switched traffic may be routed through a C&C node, but may only use point-to-point links.

A C&C node can be the origin and destination for traffic. It may also act as a switching point.

### *Editing:*

Double click on the icon to bring up the node dialog box, or highlight the icon and pick the **Edit/Detail** menu option. The name of the node, its icon, and its failure characteristics may then be defined.

Also, a parameter set must be applied to the node. The parameter set may be copied from the system library, or created locally within the model. The values found in the parameter set describe the performance characteristics of the node. (See *Parameter Sets*)

The screenshot shows the 'Processing Node Parameters' dialog box. The 'Parameter set name' is 'DEFAULT'. Under 'Application Processing', 'Processing/cycle (mic)' is 0.0 and 'Time slice (mic)' is none. Under 'Packet Processing', 'Processing/kbyte (ms)' and 'Processing/setup (ms)' are none, and 'Session limit' is 1024. Under 'Port Processing', 'Port default (ms)' is 0.000, 'Buffer max' is 100000000, and 'Buffer units' is Bytes. Under 'Circuit Switching', 'Call limit (kbps)' is 10000.000000. Under 'Disk Storage', 'Disk (Mb)' is 0.000000, 'Sector (Kb)' is 0.000, 'Xfer (mic)' is 0.0, and 'Seek (mic)' is 0.0. The 'File List' contains 'GENERAL STORAGE'. There are buttons for 'Add...', 'Edit...', and 'Remove...'. At the bottom are 'OK' and 'Cancel' buttons.

**C&C Node Parameters Dialog Box**

### *Execution:*

#### **Processor Scheduling**

The node keeps a prototype application list which is a list of all the applications that could run at the node. This list is constructed from the traffic and application sources the user connects to the node on the COMNET III user interface. A particular instance of an application will be created when the scheduling requirements for the prototype have been met.



The scheduling requirement may be by iteration time, by delay time, or by received message, depending on the type of application and the user selection. Checking and testing that scheduling criteria have been met for an application prototype does not involve any load or overhead on the modeled node processor—but it does take time on the computer on which you are running the simulation.

Time-based scheduling requirements are simply met by the application prototype picking the time the next application instance is required and then creating the instance at the required time.

Received Message Scheduling means that an application may require one or more messages to be received before a new instance of the application may be created. In this case a list of received messages must be kept so that, if several messages are required by an application, the messages are kept until all those required by the application have been received. Once all the messages are received they are then “used” by the creation of the new application instance, and so are no longer in the received message list. A particular message can only be used to start one application instance.

*(see Received Message Scheduling)*

In addition to application sources, message, response and session sources also result in an application prototype. For instance, a message source is an application source specialized to execute a single transport command.

When a new instance of an application is created from the application prototype, the new application instance is added to the end of the pending application list. When the node becomes idle it will pick the application instance at the top of this list and so becomes loaded by the commands requiring execution by the application. Since new instances are added at the end of the list, the node will inherently pick the application that has waited longest. An exception to this may occur when some applications have commands that are waiting for an ack or waiting for a file to be unlocked, they will be bypassed in favor of applications that do not have such requirements.

As well as executing applications, the node processor must also process packets from input buffers. When the node processor becomes idle it must decide whether to execute an application next or process a packet. The processor will compare the packets at the head of each input buffer and the application at the head of the application pending list. It will select the packet or application which has waited longest and perform the requisite task (packet switching or application execution) next.

Note that this algorithm does not compare priority between waiting packets in different queues—only waiting time. However, the packet at the head of each input port buffer queue will be the highest priority packet in that queue. There is no priority attribute for applications.

The packet processing task will take processor time from the node processor (assuming non-zero entries on the node parameter settings).

### **No Time Slice Execution**

If “no time slicing operation” is specified, the processor picks an application instance to execute as described above, and then executes the commands listed in the application command list in sequential order.

## 4. Modeling Constructs

The application tells the processor by name which command to execute next. The processor compares this name to the commands in its local command list and executes the command it finds which has the same name. If no command in the local list has the same name, the node looks in the global command list. Consequently local names override global names. It is a verification check that a command named in an application exists either locally or globally.

When the command completes execution, the application asks the processor to execute the next command. And so on until all of the commands have been executed. Then the processor makes a scheduling decision about which task to perform next as described above.

Message creation commands, such as transport commands, answer commands and setup commands, are exceptions to this rule. A message creation command is a requirement to generate packets and route them across the network. When the processor executes such a command, the sequence of events that occurs is that it (1) creates the message, (2) incurs the packetizing delay, (3) creates a packet, (4) processes the packet, (5) routes the packets. Steps (3) through (5) are repeated, as necessary, to packetize the entire message.

After a message has been created, a transport command will not resume execution until it has flow control clearance, so there will be no packet creation attempt until after the transport command finishes its packetizing delay. The application is only removed from the pending list if it's ready to execute, so there is no need to return it to the queue if it does not have flow control clearance.

This scheme of application scheduling has several features:

- If two message sources schedule overlapping transmissions, there will be two application instances in the pending list. The processor will take the first one, generate a packet, and then replace the application instance at the back of the pending list. It will then take the other application instance and generate a packet for it—and so on. Packets will generally be generated alternately over time for the two message sources, although they may not alternate exactly, depending on the transport layer flow control.
- If a node performs both packet processing and message generation functions, it will create or process a packet based on which generator or waiting packet has been delayed the longest. Packets moved from input to output buffers will be interleaved with newly created packets.

### With Time Slicing

If time slicing operation is specified, the basic algorithm is as outlined above but with the additional logic that, at the end of a time slice, the currently executing command of the current application instance is interrupted and the application instance is put back at the end of the pending list.

When the application instance is rescheduled later, the interrupted command is resumed where it left off and the balance of work is performed. In the case of read and write commands, the remaining number of bytes in the data transfer is completed. For a process command, the remaining number of cycles is completed. It is important to note that if a read or write command is interrupted, the file it was accessing will remain locked until the entire command is completed. This ensures file integrity in the event that some other application needs to access the same file.

Message generation commands (such as transport commands) are not interrupted by time slices. Packet creation is done one packet at a time on a round robin basis (subject to flow control constraints) for the scheduled message generators, as described in the previous section.

## Buffer Processing

A packet arrives at a node when all the frames which constitute the packet have been delivered to the node by the link protocol. The packet is reassembled from its frames and an attempt is made to insert the packet into the input port buffer which services the link.

A test is made to see whether there is sufficient space in the input port buffer to receive the packet. If there is sufficient space a test is made to see whether there is sufficient space in the overall C&C node input buffer limit. If both tests pass then the packet is placed in the port input buffer which services the link. If either test fails then the packet is considered blocked and may be retried from the origin depending on the transport protocol settings.

The packets in the buffer are ordered in decreasing priority plus FIFO order. Once in an input buffer a packet must incur a buffer processing delay before it may be considered for removal from the buffer. The buffer is modeled as owning a single delay resource which is used to incur the packet delay, one packet at a time. Consequently the packets in the input port buffer may have to wait until they are at the head of the queue in order to incur their delay.

Packets that have incurred their delay are still in the input port buffer but are now available for switching. Again these packets are ordered by decreasing priority plus FIFO. The C&C node compares the packets at the head of each of its port input buffer queues and the application at the head of the application pending list. It will schedule the item that has been waiting the longest. Assuming this is a packet at the head of a port input buffer queue, the processor will remove the packet from the queue and then incur the packet processing delay. It will then make an instantaneous routing decision to determine which output port buffer to place the packet in, and then attempt to instantaneously insert the packet into the selected port output buffer.

*(See Routing Protocol: Packet)*

If the packet is now at its destination any required acknowledgments will be created, delivery statistics updated, and the packet destroyed. If the packet is the last packet of a message, the message is now counted as having been delivered and its text can be used to satisfy received message scheduling requirements.

If the packet is being routed onwards, the attempt to insert the packet in the selected port output buffer may fail because there is insufficient space available. Again, two tests are made against the available port output buffer space and the overall C&C node output buffer space. Both tests must pass for the packet to be inserted into the selected output buffer.

If either of the buffer tests fail, then the packet is blocked, and optionally recreated and retransmitted from the originating node (depending on the transport protocol settings). This is in contrast to ATM node processing where blocked packets would remain in the input port buffer if the output port buffer was full.

Once a packet is placed in the port output buffer it must incur the buffer processing delay for the port. Similar to the input port, the packets queue in decreasing

## 4. Modeling Constructs

priority and FIFO order for the single delay resource, are delayed by the specified time, and then wait in the output buffer in decreasing priority and FIFO order to be transmitted by the downstream link.

### Command Execution

*See the appropriate command section.*

### Circuit- Switching

A C&C node may originate, switch, and receive circuit-switched calls. If a C&C node originates a call, then the destination may be the same node. This may be used if the node models a local switch which carries local traffic as well as out-of-area-traffic, but the local network topology is not desired to be modeled.

The circuit-switched traffic is independent of the packet-switched traffic. There is no impact on the port buffers or on node utilization for creating, routing or receiving circuit-switched traffic. While the one node can service all types of traffic the resources on the node are separately specified for circuit switching and packet switching.

When a circuit-switched call originates, a sequence of routing decisions is typically required, depending on the number of routes composing the end-to-end path and the number of hops in each route. The nodes and links along a possible route are inspected to determine whether the complete route has capacity to carry the call. From the point of view of an C&C node, the node has a maximum possible bandwidth which it can concurrently switch for all circuit-switched calls routed through it. As a call is routed through the C&C node, the amount of remaining bandwidth available for circuit-switched traffic is reduced. When a call clears the used bandwidth is given back.

Consequently, a C&C node is available to carry a particular call if, at the instant the routing decision is being made, the amount of free bandwidth on the C&C node is greater than (or equal to) the bandwidth requirement for the call (as defined on the routing class dialog box).

Only when the routing decision has checked that a route has sufficient capacity is the bandwidth claimed. However, it should be noted that this is an instantaneous decision.

If the C&C node is a congestion point for the circuit-switched traffic (i.e., it is a point on the route where there is insufficient bandwidth), the routing protocol will first of all test other routes to see if one is available. If no other route is available, the routing protocol may revisit the blocking node and test whether there are any calls routed through the node which may be preempted to release sufficient bandwidth to route the required call. The preemption mechanism is controlled by call priority as set on the call source dialog box and preemptions must also be activated on the call routing protocol dialog box.

*(See Routing Protocol: Call)*

### Reporting:

Node Utilization.

File Warning Report.

Buffer Reports.

All delay reports are affected by C&C node performance.

**Fields:**

<b>Name</b>	An alphanumeric field to identify the node. The name must be unique in the backbone or subnetwork.
<b>Icon</b>	The name of the icon used to represent the node.
<b>Node State</b>	Indicates whether the node is in a failed state or not. (see <i>Node</i> )
<b>Packet Routing Table</b>	(see <i>Routing Table: Packets</i> )
<b>Call Routing Table</b>	(see <i>Routing Table: Calls</i> )

**Parameter Set Fields:**

<b>Source/Sink Only</b>	This flag indicates whether the node is available for switching traffic (either packet-switched or circuit-switched). If the box is checked then the node can only be an origin or destination for traffic.
-------------------------	---

**Edit Processor...**

**Number of processors** All node types have, as part of their parameter set, a processor count, which is 1 by default. The processor count specifies the number of *processing units* available to perform work requiring processor time, for example packetizing delay or an application source's command sequence.

Nodes configured with a processor count of 1 will behave exactly as their single processor predecessors in earlier releases. The behavior of nodes configured with processor count greater than 1 depends on the setting of the time slice option.

The architecture of a processing unit includes the processor itself and two work queues. One queue holds work that can run as soon as the processor is available, and the other holds work that is suspended until some conditions, such as the arrival of messages to satisfy a filter command, are met. In addition, the collec-

## 4. Modeling Constructs

tion of processing units share a single pair of queues. These shared queues serve the same purpose as their cousins on the processing units, but the shared queues hold work that has not yet been assigned to a processing unit.

A key aspect of COMNET's multiprocessor model is that, once work has been assigned to a processing unit, it remains on that processing unit (in either the processor itself, or one of the two local work queues) until it's finished.

When time slicing is enabled, at the beginning of each time slice, work that has not yet been assigned to a processing unit is selected, assigned to a processing unit, and given processor time. Whenever there is no unassigned work, processor time is shared among the processes of a processing unit in a same round robin fashion

When time slicing is not enabled, work is assigned to available processors until all processors have active work, at which point the processors continue to process their work until finished. When a processor becomes available, because all its work is either finished or suspended, it selects work from one of the shared work queues using the same algorithm as that used to select work from local queues.

An application's command sequence makes up a single item of work, and will be assigned to one processing unit.

### Selection Rules

The node's processor has options for how it selects the next application or packet to process after it finishes processing the current application or packet. These options are important when modeling heavily congested processors that will frequently have many choices for selecting the next task. The rules for selecting the next task can have significant effects on the delay of applications or messages.

The application and packet selection options are available in the processor parameters that are part of the node's parameter set.

### Between applications

There are the following selection options for when the processor selects an application from among multiple ready applications in the application queue:

#### First App Available

This option selects the application ready for processing and that is closest to the front of the application queue. After the application is done processing it is placed at the back of the queue so that this approximates a round robin processing of applications that are ready. Applications that are not ready for processing will tend migrate to the front of the queue and thus it will get preference as soon as the application becomes available.

#### App Ready The Longest

This option searches the full application queue for the application that has been ready for processing the longest.

### Between Apps and Packets

The processor may have different criteria for picking between packets waiting at input buffers and applications waiting in the application queue. The options include the following choices:

#### Pkts Have Priority

For this option, the processor will process any packet that is ready at an input buffer before processing any applications waiting at the application queue. This type of processing places more importance on routing than on processing and can be useful for modeling networking devices. This may also be useful for modeling

cases where the arrival of a packet triggers an interrupt that make the processor process the incoming packet immediately after finishing the current task.

### **Apps Have Priority**

For this option, the processor will process any waiting application before it will process any packet waiting at an input buffer. This can be useful for modeling simpler devices that do not frequently check for waiting packets nor use interrupts for packets. It can also be useful for modeling complex scenarios where new packets will give the processor new tasks and it is desired to have the processor finish with the current task before introducing a new one.

### **Earliest Arrival**

This option selects the application or packet that arrived at the node at the earliest time. The packet arrives at the node when it enters the input buffer. The application arrives at the node when it is generated or triggered. This is the behavior that was modeled in earlier releases.

Note that an application or packet may not be ready for processing at the time it arrives at a node: the application may wait for one of a number of conditions to occur before being ready to process, and the packet may require port processing before it is ready to be processed by the node. In either case, the packet or application will not be processed until they are ready, but when they are ready, they will be picked based on which one arrived at the node the earliest.

### **Ready The Longest**

This option is similar to the earliest arrival option except that the time that the application is ready to run is compared with the time that the packet arrived at the input buffer. The applications ready-to-run time may be substantially more recent than its arrival time in the queue. Examples of when an application's ready to run time may be set are when:

- 1) a transport command has to retransmit a blocked packet
- 2) a transport command receives an acknowledgement to advance its flow control window
- 3) a transport command is allowed to transmit by its rate control algorithm
- 4) a filter command is satisfied
- 5) a file is unlocked for a waiting read or write command

Note that the packet's time that is compared is still the time that the packet arrives at the buffer instead of when it is ready after being processed and arriving at the front of the buffer. Generally this should be an insignificant difference compared with the difference between the applications' ready-to-run time and arrival time.

### **Between pkt buffers**

There are the following selection options for when the processor selects a packet to process from among multiple input buffers:

#### **Earliest Arrival**

This option searches all the buffers for the packet that arrived the earliest.

#### **Packet Priority**

This option only considers the packets at the front of the buffer and thus will preserve the sorting within the buffer. If multiple buffers have packets waiting, the highest priority packet will be selected, and if there are multiple head-of-line packets with the same priority, the processor will select the packet that arrived at

## 4. Modeling Constructs

its buffer the earliest.

### Processor Usage

COMNET III makes the utilization of the node's processor an option for many of the modeled delays. If the delay is modeled with processor utilization, then the items will have to wait for an available processor and then process sequentially until finished. Delays without utilization can allow many such delays to occur simultaneously and, just as importantly, without affecting the utilization of the node. A delay without processor utilization is useful for modeling delays on resources that are not part of the model (such as human delays or processors that are outside of the model) or they may be used for waiting delays unrelated to any sequential processor.

### Packet processing uses processor

The packet processing parameters (for modeling network-layer processing at the node) may optionally utilize the processor. This option is also in the processor parameters that are part of the node's parameters. It is useful for modeling devices with complex multiprocessor or switching fabrics that allow most delays to occur simultaneously.

### Read/write commands use processor

The storage delay for read and write commands may optionally utilize the processor. The option to utilize the processor for storage delays is in the processor parameters that are part of the node's parameter set. The primary intent here is to model the separate disk controller for doing the main processing of the storage device without utilizing the main processor. This allows the main processor to process other activities while the disk controller delay is in progress.

Note that if multiple commands access the same file simultaneously, COMNET III will model the accesses sequentially even when the delay does not utilize the central processor because when one command begins, it locks the file so that other commands have to wait until the file is unlocked before they can access it.

### Processing Time/Cycle

Processing command execution times are scaled in cycles. The actual duration of a processing command is the number of cycles multiplied by the node's processing time per cycle. When the processing command is executed, the node processor will be made busy for this time period.

If, for example, a C&C node is modeling a server, then it may have several different process commands that are called from several different applications. If one of the goals of a simulation is to investigate the effect of server processor speed on network performance, the processing time per cycle field allows the speed to be ramped up or down by changing only the one parameter (rather than changing all the process commands individually). Note that the processing time per cycle field only affects the speed of execution of process commands. Read and write command durations are controlled by the disk performance characteristics, and the packet switching delays are controlled by the buffer and packet performance characteristics.

If a server runs at 50MHz, the processing time need not be entered as .02 microseconds. It may be more convenient to choose a processing time per cycle to be a convenient unit for the processing delays modeled at this node, and perhaps normalized to a unit value for the default processor being modeled. For example if server commands take on the order of milliseconds to complete, then a conven-



ient processing time per cycle would be 1 millisecond. Faster or slower machines may be modeled by adjusting that time smaller or larger, respectively. For example, a 10% faster machine would have 0.9 milliseconds per cycle.

### Time Slice

If no time slice value is entered then COMNET III assumes no time slice operation for the node.

If a value is entered then, as commands are executed, a particular command can use the node processor for one time slice duration and then must give up the processor to other pending applications.

For commands which create packets, a packet is not created until after the packetizing elapses, as specified on the respective message source dialog box incurred by the node processor. The packetizing delay is not subject to time slicing.

Likewise, packet processing delays are not subject to time slicing.

Operation of the node with and without time slicing is discussed in the Execution section above.

### Processing Time/Packet

After a packet is created or after a packet leaves an input buffer, the node processor is made busy for the Processing Time/Packet value depending on the routed protocol ID of the transport layer that created the packet.

For instance, if a C&C node is modeling a packet switch rated at switching 100 packets per second, this performance could be captured by entering a Processing Time/Packet of 10 milliseconds as the default or for this particular routed protocol ID. Upgrading the switch to a 500 packet per second rating would then imply reducing the Processing Time/Packet to 2 milliseconds.

The Processing Time/Packet is a delay applied to each packet as it is switched through the node. It is in addition to any input or output buffer queueing delays or buffer processing delays.

After incurring a packetizing delay, packets would be created at a node and this would then be followed by the packet processing delay, then the packets would be inserted in an output port buffer.

Packets destined for the node would first incur any input port buffer switching and queueing delays, followed by this packet processing delay based on its routed protocol ID. They are then counted as having arrived.

The packet processing time at a Computer and Communication Node, Computer Group Node, and Router Node can vary according to a packet's routed protocol. The packet's routed protocol is determined by the transport protocol referenced by the packet. A transport protocol has a new field identifying its routed protocol. In setting the routed protocol field, the user may select one of the names in the combo box list or type in a new name.

When a packet is about to be processed at a node, the packet processing time is determined by going to the node's list of packet processing times and looking for a routed protocol in the list that matches the packet's routed protocol. If no match is found, the processing time for the routed protocol named DEFAULT is

## 4. Modeling Constructs

used. The DEFAULT entry is always present in the list. The total processing time for a packet is given by the protocol-dependent processing time per packet (as just described) plus a separate factor that varies linearly according to the size of the packet. The size-dependent factor is given by the processing time per kilobyte field times the packet size in kilobytes. Session setup packets are treated as a special case and have a packet processing time given by the processing time per setup packet field.

The list of packet processing times for each routed protocol is accessed by clicking on the Edit Times button in the Packet Processing part of the Parameters dialog box for C&C Nodes, Computer Groups, and Routers.

*(see also Processing Time/kbyte following)*

### Processing Time/kbyte

Some devices exhibit the behavior where it takes longer to switch a large packet than a small packet. The C&C node makes allowance for this by providing a Processing Time/kbyte field.

The switching time for a packet is then calculated using the formula:

$$\text{Switching Time} = Ax + B$$

where

A = Processing Time/kbyte

x = Size of packet in kilobytes

B = Processing Time/Packet (see immediately previous section)

### Processing Time/Setup Packet

When a session is established across a network, the session setup packet has to be routed at each node. Assuming connection-oriented routing, data packets that subsequently follow the same route as the session setup packet do not incur the same routing overhead at the node, because as the route has already been established by the setup packet. Thus the processing for the setup packet may be substantially longer than the packets that follow in that session.

Consequently, COMNET III provides the Processing Time/Setup Packet to be entered for the setup packet delay at the node. The Packet Processing Time/ Packet delay will be used for the following data packets.

### Input Buffer Size

The maximum allowable buffer usage across all input ports connected to the C&C node. When a packet is received, then a test is first made to see if there is space in the port buffer, and then a test is made to see whether the total buffer usage limit (as defined here) would be exceeded if the packet were accepted.

The input port buffer size is defined on the arc which connects a particular link to the C&C node.

If there is insufficient space to receive a packet then the packet is blocked and optionally retried depending on the transport protocol settings.

### Output Buffer Size

The maximum allowable buffer usage across all output ports connected to the C&C node. When a packet is switched to an output port, then a test is first made to see if there is space in the port buffer, and then a test is made to see whether

the total buffer usage limit (as defined here) would be exceeded if the packet were accepted.

The output port buffer size is defined on the arc which connects a particular link to the C&C node.

If there is insufficient space to receive a packet into the output port buffer, then the packet is blocked and optionally retried from the origin depending on the transport protocol settings.

### Session Limit

The maximum number of sessions that can be concurrently routed through the C&C node at any particular instant.

When a session routing decision is being made for a session setup packet, C&C nodes that are already carrying a number of sessions equal to the session limit are not available for routing the session setup packet.

(see *Routing Protocol: Packet*)

### Port Processing Times

Port processing times can vary as a function of a packet's routed protocol and the type of link connected to the port. To determine the port processing time for a packet that has just arrived at an input or output port buffer, the model first checks the special case port processing times included with the node's parameters. It searches the list of special cases to find a case with the same routed protocol and type of link. If a match is found, the model uses the port processing time given for the special case. If no match is found, it uses the port processing time set for the port. The processing time for the port is set on the arc connecting the node to the link. When a node is initially connected to a link, the processing time that is set for the port is the default port processing time on the node parameters dialog box. The port processing time set on a port can subsequently be changed by editing detail on the arc connecting the node and link.

To edit the list of special case port processing times, click on Edit Times in the Port Processing part of the node parameters dialog box. Note that protocol and link-dependent port processing times and default port processing times apply to any type of COMNET III node. As explained above, the default port processing time on the node parameters dialog box is used only at edit time when a node is connected to a link. Including the default port processing time with the node parameters allows port-dependent processing times to be saved in a library along with the other parameters describing a node.

### Call Limit

The maximum amount of bandwidth for circuit-switched traffic that can be concurrently routed through the C&C node at any particular instant.

When a circuit-switched routing decision is being made for a call, C&C nodes that have insufficient remaining bandwidth to carry the call are not available for routing the call.

(see *Routing Protocol: Call*)

### Disk Size

If read and write commands are required on the C&C node, the size of the modeled disk connected to the node must be specified.

The value is entered in Megabytes (1024 Kbytes = 1024 \* 1024 bytes = 1,048,576 bytes)

## 4. Modeling Constructs

If a write command adds data to a file which results in the total space for all files exceeding the specified disk size, a warning will be added to the File Warning report and the command will continue as if the full transfer completed, although, the disk will not be increased in size. This is taken as a pessimistic response to overflowing the disk.

*(see also Command: Write File & Command: Read File)*

### Disk Sector Size

The size of a disk sector in Kilobytes (1024 bytes).

A read or write command will result in a number of disk sectors to be read or written, depending on the amount of data and the sector size.

### Disk Transfer Time

The length of time it takes to read or write a disk sector.

### Disk Transfer Overhead

The overhead time to seek to a particular disk sector.

The duration of any particular transfer is calculated on the basis of:

$$\text{Transfer Time} = \text{Number of Sectors} * (\text{Transfer Time} + \text{Overhead Time})$$

*(see also Command: Write File & Command: Read File)*

### File List

A list of files that are to be created on the modeled disk before the simulation starts. Only files with a positive size are considered to have been created.

Files may also be created by executing a write command. If the named file does not exist, then the write command will create it, provided the command appends a positive number of bytes to the file.

However, it is common to need a number of files on the disk and so these can be added to the file list, rather than having to create and execute a write command for each file. When the model initializes, the listed files are created on the disk, provided they have a positive size.

It may be that at the time the simulation is being performed the detailed file structure of the modeled system is not known. In this case COMNET III provides a file called GENERAL STORAGE which can be used to read and write arbitrary numbers of bytes, thereby modelling the transfer delays but without knowing the detailed file structure.

### File Name

The name of the file. The name must be unique on the C&C node.

### File Size

The initial size of the file. It is a verification check that the size of all initial files does not exceed the size of the disk.

### File Read Only Flag

Indicates whether the file is read-only or not. Attempting to update a read-only file will cause an error to be reported when verifying a model.

### Command List

Answer Message Command *(see Command: Answer Message)*

Processing Command *(see Command: Processing)*

Read File Command *(see Command: Read File)*

Setup Session Command *(see Command: Setup Session)*

Transport Command (*See Command: Transport*)

Write File Command (*see Command: Write File*)

## 4. Modeling Constructs

### 4.59 Node: Computer Group

#### *Purpose:*

It is a common requirement to want to model a population of similar users connected to a network. Rather than making a node to represent each of them, and then editing the details of each node, a Computer Group Node only has to be defined once to represent a number of end systems.

In many respects, the Computer Group Node is the same as a C&C node. It has largely the same parameters and performance characteristics, as evidenced by the user interface. However, the Computer Group node has a Quantity attribute which indicates how many identical instances of the node to generate with the Computer Group node parameters. Hereafter, one of the individual nodes represented by the computer group is referred to as an *Instance Node*.

Each Instance Node may execute applications, generate packets, and receive packets, as if it were a C&C Node. However, it may not switch or route packets.

#### *Creating:*

To create a Computer Group node, pick the appropriate node tool from the palette and drag a node onto the background. Or use the create menu option to create a Computer Group node. On the dialog box accessed via double clicking on the icon, you can indicate in the type box to use Computer Group operation if the node you have created is of a different type. Once the node has been created a parameter set must be applied to it via a selection in the Parameter pulldown box. (See *Parameter Sets*)

#### *Connectivity:*

A Computer Group can connect to just one of the multiaccess links. Point to point links may not be connected to such a node. Note that a Computer Group node represents a population of Instance Nodes, but a point to point link may only connect to one node at either end. This explains the restriction on point-to-point link connectivity.

Each Instance Node in the Computer Group has an input port buffer and an output port buffer for the link it is connected to. The details of these buffers may be accessed via double clicking on the arc which connects the link to the Computer Group node.

Circuit-switched traffic cannot be created, routed through, or received by a Computer Group node (or any of the Instance Nodes). Circuit-switched traffic can only be carried over point-to-point links. Since a point-to-point link cannot connect to a Computer Group node, the Computer Group node can have no circuit switching capability.

Multiple Message, Session, Response and Application sources may be connected to a Computer Group node. A source connected to a Computer Group Node implies that there is a separate source for each Instance Node in that group.

#### *Editing:*

Double click on the icon to bring up the node dialog box, or highlight the icon and pick the **Edit/Detail** menu option. The name of the node, its icon, and its failure characteristics may then be defined.

Also a parameter set must be applied to the node. The parameter set may be copied from the system library, or created locally within the model. The values found in the parameter set describe the performance characteristics of the node. (See *Parameter Sets*)

**Computer Group Node Parameters Dialog Box**

**Execution:**

(See Node: Computer & Communications for the main discussion)

When the simulation starts, as many Instance Nodes as indicated by the Quantity field are created. Each Instance Node has its own copy of connections to the multiaccess links, and its own copy of any sources connected to the Computer Group node.

Thereafter, each Instance Node operates independently of the other Instance Nodes. It will generate traffic, access links and make routing decisions just as if it were an independent C&C node. However, the following points and observations should be noted:

Each Instance Node has its own copy of the traffic sources that were connected to the Computer Group node. For example, maybe the Computer Group node on the user interface has a message source connected to it with a mean interarrival rate of exponential(10) seconds, and a Quantity value of 10. Therefore, there are 10 Instance Nodes and 10 message sources (one for each node). If no start time is indicated on the message source, each of the sources will pick an independent random time between 0 and 10 for the first message to be created. Consequently, all Instance Nodes will create traffic on the desired profile, but each will pull its generation time independently of the others.

If a message has a destination in a Random List, Weighted List, or Random Neighbor List which is a Computer Group node, the message will only be delivered to one of the Instance Nodes in the Computer Group. A uniform statistical pick between 1 and the number of Instance Nodes in the group will be made and the “lucky” Instance Node selected to receive the message.

If the destination is a Multicast List, all Instance Nodes in the Computer Group

#### 4. Modeling Constructs

will receive the message.

***Reporting:***

As for a C&C node.

On the Report Selection screen you may indicate whether Computer Group Detail should be reported. If this is off, then only sum total statistics across all Instance Nodes in a Computer Group are reported. If this is on, each Instance Node in the Computer Group is reported.

***Fields:***

A Computer Group Node has the same fields as a C&C Node, with the following exceptions:

**Quantity**

The number of identical C&C nodes represented by the Computer Group Node. Individual copies of the node are made up to this quantity. These are referred to as Instance Nodes. Each Instance Node has connections to any links the Computer Group Node is connected to. Connected traffic generators are copied for each Instance Node.

**Call Bandwidth**

Not Supported

Circuit-switched traffic cannot originate, be switched through, or be destined for a Computer Group Node.

**Source/Sink Only**

Not necessary because computer group instances are always source/sink only.

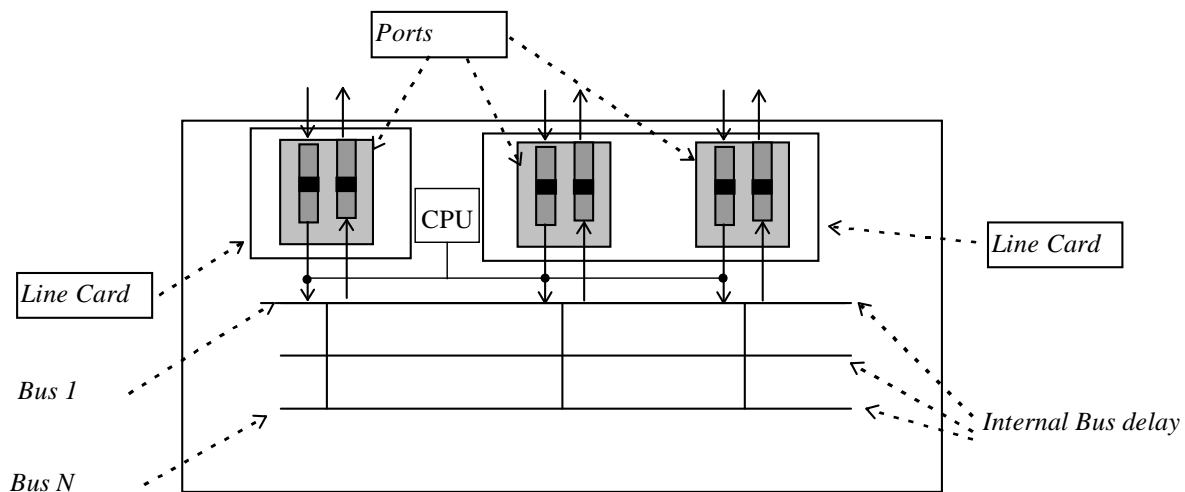


## 4.60 Node: Router

### *Purpose:*

A Router Node is aimed at modelling router-based networks in packet-switched environments. The basic premise is that for a Router Node, the performance can be effectively captured via its input and output buffering characteristics and internal switching rate. Performance characteristics arising from internal design architecture, software release numbers, etc., can be aggregated into the switching speed parameters.

Consequently, in COMNET III a Router node has input and output ports which have associated buffer sizes and buffer processing times, and an internal bus for moving packets from port to port. The line card on which the port is located may be defined—this will determine whether a bus delay is required or not. In addition, the router has the processing modeling capabilities of a Computer & Communications Node. Processing from the central processor is modeled after the packet leaves a buffer but before it crosses the bus (if required).



**Router Node Architecture**

### *Creating:*

To create a Router node, pick the appropriate node tool from the palette and drag a node onto the background. Or use the **Create** menu option to create a Router node. On the dialog box accessed via double clicking on the icon, you can indicate in the type box to use Router operation if the node you have created is of a different type. Once the node has been created a parameter set must be applied to it via a selection in the Parameter pulldown box. (See *Parameter Sets*)

### *Connectivity:*

All types of links may connect to a Router node. As many connections as desired may be made. For each link connected to the node there is an input port buffer and an output port buffer. The details of these buffers may be accessed via double clicking on the arc which connects the link to the node.

The port details may also include a Line Card ID field. Ports which have the same non-blank Line Card ID value are assumed to be connected to the same

#### 4. Modeling Constructs

Line Card in the Router. Transfers between such line cards will not have to be via a bus and so will not incur a bus delay. Ports with different Line Card IDs and ports with no Line Card ID are connected internally via transmission across the internal Router bus. Transmissions between such ports incur bus transmission delays.

Circuit-switched traffic may be routed through a Router node, but may only use point-to-point links. Circuit-switched traffic does not use the ports or the bus.

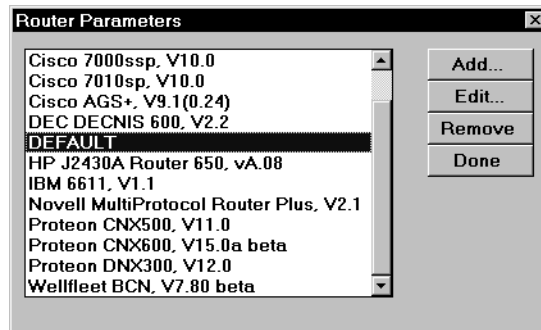
A Router node cannot be an origin or a destination for traffic. It may only act as a switching point.

#### *Editing:*

Double click on the icon to bring up the node dialog box, or highlight the icon and pick the **Edit/Detail** menu option. The name of the node, its icon, and its failure characteristics may then be defined.

Also a parameter set must be applied to the node. The parameter set may be chosen from the model's list. The values found in the parameter set describe the performance characteristics of the node. Many vendor-specific parameter sets are available. These parameters are based on test data from the Harvard Network Device Test Lab.

*(See Parameter Sets)*



**Standard Router Choices**

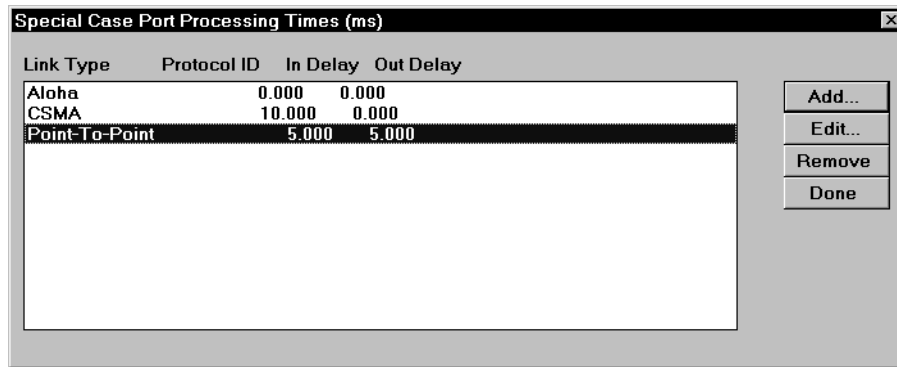
**Router Node Parameter Set Dialog Box**

Double clicking on the arc which connects a link to the Router brings up the port characteristics. These include the port buffer sizes, port delays, line card id, and routing penalty tables to use.

**Port Parameter Dialog Box**

You can also have a special processing time depending on which port of the Router the packet is entering. This can be accessed by clicking on Edit Times... in the Port Processing section of the Router Parameters dialog box. Each type of link, such as Aloha or CSMA, can have a different input and output port delays

## 4. Modeling Constructs



**Special Case Port Processing Times Dialog box**

### *Execution:*

#### **Packet Switching**

A packet arrives at a node when all the frames which constitute the packet have been delivered to the node by the link protocol. The packet is reassembled from its frames and an attempt is made to insert the packet into the input port buffer which services the link.

A test is made to see whether there is sufficient space in the input port buffer to receive the packet. If there is sufficient space a test is made to see whether there is sufficient space in the overall Router node input buffer limit. If both tests pass then the packet is placed in the port input buffer which services the link. If either test fails then the packet is considered blocked and may be retried from the origin depending on the transport protocol settings.

The packet occupies buffer space while it waits its turn to be processed at the port. Once it has finished processing, it continues to occupy input buffer space while it waits its turn to be processed by the node processor. After it is processed by the node, a routing decision is made that determines which output port the packet should use (assuming the packet is not yet at its destination).

The packets in the buffer are ordered in decreasing priority plus FIFO order. Once in an input buffer, a packet must incur a buffer processing delay before it may be considered for removal from the buffer. The buffer is modeled as owning a single delay resource which is used to incur the packet delay, one packet at a time. Consequently, the packets in the input port buffer may have to wait until they are at the head of the queue in order to incur their delay.

Packets that have incurred their delay are still in the input port buffer but are now available for switching. Again these packets are ordered by decreasing priority plus FIFO. The Router node will then take the packet at the head of the buffer, make an instantaneous routing decision to determine which output port buffer to send it to, and then attempt to move the packet from the port input buffer to the selected port output buffer.

*(See Routing Protocol: Packet)*

If the line card ID of the output port is different from the line card ID of the input port, or the line card ID's are not specified, the Router assumes that transmission

across an internal bus is required. At the point the routing decision is made, no bus may be available so the packet has to remain in the input port buffer until a bus is available. A transmission time across the bus will then be computed from the packet size and bus transmission speed, the packet removed from the input port, the delay time incurred, and an attempt made to insert the packet in the output port buffer.

If no bus transmission is required, then the packet will be removed from the input port buffer and an attempt made to insert it in the output port buffer.

In either case, the attempt to insert the packet into the output port buffer may fail because the selected buffer has insufficient space available. Again, two tests are made against the available port output buffer space and the overall Router node output buffer space. Both tests must pass for the packet to be inserted into the selected output buffer.

If either of the buffer tests fail then the packet is blocked and is either discarded or retransmitted on an end-to-end basis, depending on the selected transport protocol option. This is in contrast to ATM node processing, where blocked packets would remain in the input port buffer.

When a bus becomes idle, it pulls the packet that has been waiting the longest

Once a packet is placed in the port output buffer, it must incur the buffer processing delay for the port. Similar to the input port, the packets queue in decreasing priority and FIFO order for the single delay resource, are delayed by the specified time, and then wait in the output buffer in decreasing priority and FIFO order to be transmitted by the downstream link.

## **Circuit-Switching**

When a circuit-switched call originates, a sequence of routing decisions is typically required, depending on the number of routes composing the end-to-end path and the number of hops in each route. The nodes and links along a possible route are inspected to determine whether the complete route has capacity to carry the call. From the point of view of a Router node, the node has a maximum possible bandwidth which it can concurrently switch for all circuit-switched calls routed through it. As a call is routed through the Router node, the amount of remaining bandwidth available for circuit-switched traffic is reduced. When a call clears the used bandwidth is given back.

Consequently, a Router node is available to carry a particular call if, at the instant the routing decision is being made, the amount of free bandwidth on the Router node is greater than (or equal to) the bandwidth requirement for the call (as defined on the routing class dialog box).

Only when the routing decision has checked that a route has sufficient capacity is the bandwidth claimed. However, it should be noted that this is an instantaneous decision.

If the Router node is a congestion point for the circuit-switched traffic (i.e., it is a point on the route where there is insufficient bandwidth), the routing protocol will first of all test other routes to see if one is available. If no other route is available, the routing protocol may revisit the blocking node and test whether there are any calls routed through the node which may be preempted to release sufficient bandwidth to route the required call. The preemption mechanism is control-

## 4. Modeling Constructs

led by call priority as set on the call source dialog box and preemptions must also be activated on the call routing protocol dialog box.  
(See *Routing Protocol: Call*)

### **Reporting**

The buffer reports indicate buffer congestion on the Router node. See:

- Input Buffer Report By Port
- Input Buffer Report By Node
- Output Buffer Report By Port
- Output Buffer Report By Node

The Message Delay and Packet Delay reports reflect delays at the Router node.

### **Fields:**

**Name** An alphanumeric field to identify the node. The name must be unique in the backbone or subnetwork.

**Icon** The name of the icon used to represent the node.

**Node State** Indicates whether the node is in a failed state or not.  
(see *Node*)

**Packet Routing Table** (see *Routing Table: Packets*)

**Call Routing Table** (see *Routing Table: Calls*)

### **Parameter Set Fields:**

**Processing Time/Cycle** Processing command execution times are scaled in cycles. The actual duration of the command is the number of cycles multiplied by the processing time per cycle. When the processing command is executed, the node processor will be made busy for this time period.

Processing commands are useful for modeling internal processing overhead at the router's CPU, such as for routing table calculations or management functions

**Time Slice** If no time slice value is entered then COMNET III assumes no time slice operation for the node.

If a value is entered then, as commands are executed, a particular command can use the node processor for one time slice duration and then must give up the processor to other pending applications.

For commands which create packets, a packet is not created until after the packetizing elapses, as specified on the respective message source dialog box incurred by the node processor. The packetizing delay is not subject to time slicing.

Operation of the node with and without time slicing is discussed in the Execution section above.

**Processing Time/Packet** After a packet is created or after a packet leaves an input buffer, the node processor is made busy for the Processing Time/Packet value.

For instance, if a C&C node is modeling a packet switch rated at switching 100 packets per second, this performance could be captured by entering a Processing Time/Packet of 10 milliseconds. Upgrading the switch to a 500 packet per second rating would then imply reducing the Processing Time/Packet to 2 milliseconds.

The Processing Time/Packet is a delay applied to each packet as it is switched through the node. It is in addition to any input or output buffer queueing delays or buffer processing delays.

After incurring a packetizing delay, packets would be created at a node and this would then be followed by a processing delay, then the packets would be inserted in an output port buffer.

Packets destined for the node would first incur any input port buffer switching and queueing delays, followed by this processing delay. They are then counted as having arrived.

The packet processing time at a Router Node can vary according to a packet's routed protocol. The packet's routed protocol is determined by the transport protocol referenced by the packet. A transport protocol has a new field identifying its routed protocol. In setting the routed protocol field, the user may select one of the names in the combo box list or type in a new name.

When a packet is about to be processed at a node, the packet processing time is determined by going to the node's list of packet processing times and looking for a routed protocol in the list that matches the packet's routed protocol. If no match is found, the processing time for the routed protocol named DEFAULT is used. The DEFAULT entry is always present in the list. The total processing time for a packet is given by the protocol-dependent processing time per packet (as just described) plus a separate factor that varies linearly according to the size of the packet. The size-dependent factor is given by the processing time per kilobyte field times the packet size in kilobytes. Session setup packets are treated as a special case and have a packet processing time given by the processing time per setup packet field.

The list of packet processing times for each routed protocol is accessed by clicking on the Edit Times button in the Packet Processing part of the Parameters dialog box for C&C Nodes, Computer Groups, and Routers.

*(see also Processing Time/kbyte following)*

### **Processing Time/kbyte**

Some devices exhibit the behavior where it takes longer to switch a large packet than a small packet. The router node makes allowance for this by providing a Processing Time/kbyte field.

The switching time for a packet is then calculated using the formula:

$$\text{Switching Time} = Ax + B$$

where

A = Processing Time/kbyte

x = Size of packet in kilobytes

B = Processing Time/Packet (see immediately previous section)

## 4. Modeling Constructs

### Processing Time/Setup Packet

When a session is established across a network, the session setup packet has to be routed at each node. Assuming connection-oriented routing, data packets that subsequently follow the same route as the session setup packet do not incur the same routing overhead at the node, because as the route has already been established by the setup packet.

Consequently, COMNET III provides the Processing Time/Setup Packet to be entered for the setup packet delay at the node. The Processing Time/Packet delay will be used for the following data packets.

### Port Processing Times

Port processing times can vary as a function of a packet's routed protocol and the type of link connected to the port. To determine the port processing time for a packet that has just arrived at an input or output port buffer, the model first checks the special case port processing times included with the node's parameters. It searches the list of special cases to find a case with the same routed protocol and type of link. If a match is found, the model uses the port processing time given for the special case. If no match is found, it uses the port processing time set for the port. The processing time for the port is set on the arc connecting the node to the link. When a node is initially connected to a link, the processing time that is set for the port is the default port processing time on the node parameters dialog box. The port processing time set on a port can subsequently be changed by editing detail on the arc connecting the node and link.

To edit the list of special case port processing times, click on Edit Times in the Port Processing part of the node parameters dialog box. Note that protocol and link-dependent port processing times and default port processing times apply to any type of COMNET III node. As explained above, the default port processing time on the node parameters dialog box is used only at edit time when a node is connected to a link. Including the default port processing time with the node parameters allows port-dependent processing times to be saved in a library along with the other parameters describing a node.

### Bus Rate

The bus transmission speed inside the node. Packets that have to be moved between ports that have different (or no) line card ID have to be transmitted across a bus and so incur a transmission delay based on the packet size and bus rate. A bus can transmit one packet at a time.

In some router implementations, a packet may need to cross a bus multiple times (for example, once from input to central processor and then again to get to output). In these cases, the bus rate should represent the effective rate as if that same packet only crossed once.

### Bus Count

Routers or switches able to switch a number of packets simultaneously, have that number as bus count. Also, nonblocking switch fabrics may be modeled where bus count equals or exceeds the number of connected links (or line cards).

### Input Buffer Limit

The maximum allowable input buffer usage across all input ports connected to the Router node. When a packet is received then a test is first made to see if there is space in the port buffer, and then a test is made to see whether the total buffer usage limit (as defined here) would be exceeded if the packet were accepted.

The input port buffer size is defined on the arc which connects a particular link to



the Router node.

If there is insufficient space to receive a packet then the packet is blocked and optionally retried depending on the transport protocol settings.

### **Output Buffer Limit**

The maximum allowable output buffer usage across all output ports connected to the Router node. When a packet is switched to an output port then a test is first made to see if there is space in the port buffer, and then a test is made to see whether the total buffer usage limit (as defined here) would be exceeded if the packet were accepted.

The output port buffer size is defined on the arc which connects a particular link to the Router node.

If there is insufficient space to receive a packet into the output port buffer, then the packet is blocked and is optionally retransmitted from the origin depending on the Transport protocol settings.

## 4. Modeling Constructs

### 4.61 Node: Router - Updated Router Library

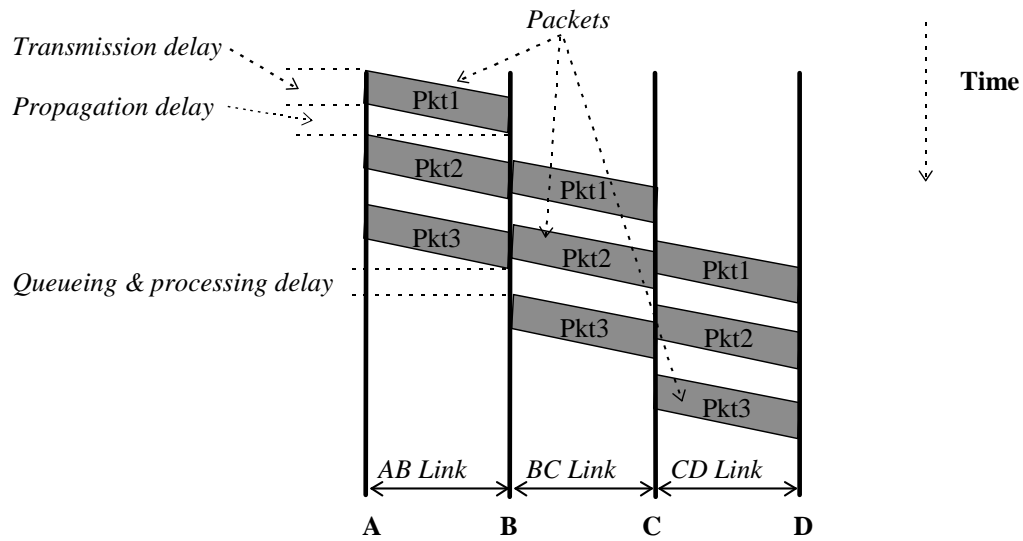
The router library has been updated to include Bradner test results published as of July 1996, as well as additional routers from Bay Networks and Cisco. Parameter sets are now available for the following routers:

- 3Com NetBuilder II
- 3Com ONcore
- ACC ACS 4200, V5.0.3
- Bay Networks BCN
- Bay Networks BLN
- Bay Networks ASN (1 unit)
- Bay Networks ASN (2 units)
- Bay Networks ASN (3 units)
- Bay Networks ASN (4 units)
- Cabletron ATX LAN Switch
- Cisco AGS+
- Cisco 2500
- Cisco 4000
- Cisco 4500
- Cisco 7000/7010 sp
- Cisco 7000/7010 ssp
- Cisco 7500
- Cray Communications ES-1520
- DEC DECNIS 600, V2.2
- Digital RouteAbout Central EW
- FORE Systems PowerHub 7000
- FORE Systems PowerHub 3150
- HP AdvanceStack Router 650
- IBM 2210 Nways Router 24E
- IBM Nways 6611
- Novell MultiProtocol Router Plus, V2.1
- Proteon CNX500, V11.0
- Proteon CNX600, V15.0a beta
- Proteon DNX300, V12.0
- RAD Network Devices Vgate
- Retix rx7500

## 4.62 Packet Switched Networks

- Purpose:** One of the main aims of COMNET III is to model packet-switched networks. Examples include X.25 networks, router-based networks running under TCP/IP, etc. Message-switched networks may also be modeled where messages are routed on a store-and-forward basis to the destination. Likewise, frame or cell switched networks may be modeled where the packet models the frame or cell.
- Creating:** There are three main steps in building a packet-switched model.
- σ Define the network architecture in terms of nodes, links, routers etc. Specify the performance characteristics of each device.
  - σ Define the traffic load to be carried between the different origin/destination combinations in the network. This may be broken down by type of traffic (E-mail, file transfer, database query, etc.), and by transport protocol.
  - σ Define the routing protocol which will make the routing decisions to move the traffic from origin to destination.
- Connectivity:** Packet-switched traffic originates, is switched by, and is destined for either Computer & Communications Nodes or Computer Group Nodes. All nodes may switch packets.
- All types of links may carry packets.
- Editing:** Via the appropriate object dialog box.
- Execution:** This description is a high level overview of packet switching. For the detailed execution logic at a node or link, see the appropriate section in this manual.
- Traffic load is defined in terms of message interarrival time and message size. This causes messages to be created at the appropriate time at the originating node. The message generator has a transport protocol associated with it that defines the maximum packet size and packet overhead (i.e., size of header, CRC check, etc.).
- Therefore packets are created one after the other until sufficient packets have been made to match the overall size of the message. Each packet is then routed to its destination by the network and associated routing protocol.
- With session-type traffic and connection-oriented routing, a session setup packet establishes the route to the destination—all subsequent data and acknowledgment packets follow the established route.
- If connection-oriented routing is turned off then all packets are individually routed for all types of message sources. This means that some packets in the same message may take a different route from others in order to reach the destination.

#### 4. Modeling Constructs



**Packet Transmission Diagram**

**Reporting:**

Message Delay Report.  
Packet Delay Report.  
Node, Link & Buffer Reports reflect packet loading.  
Session reports.

**Fields:**

See the appropriate object section.

## 4.63 Parameter Sets

*Purpose:*

Each node and link type in COMNET III has parameters associated with it which define the performance characteristics of the object. For instance, a Router object has parameters for input buffer size, output buffer size, bus rate, and bus count.

Rather than each node or link having its own local parameter values, the object points to a stand-alone set of values that may be shared among many objects. This means that where objects share common parameter values, any specific value has to only be updated once to make the change known to all the objects.

Parameter sets for each node and link type are kept in a system library which is accessible by all models when they are created. The library parameter sets are copied when a new model is made and become local to the created model. They can then be used by the objects in the model. If a new parameters set is added to the system library after a model has been created, provision is made to add this new parameter set to a particular model via the Add button which appears on the parameter set list box.

As local values of the parameters in a model's parameter set are changed (via the Edit button on the parameter set list box) then the nodes or links, which reference model's modified parameter set see the changed values.

In addition to its lists of node and link parameter sets, a model also has the following lists:

- Call Routing Classes
- Packet Routing Classes
- Transport Protocols
- Packet Routing Penalties
- Call Routing Penalties
- User Distributions
- Table Distributions

Each of these model lists is analogous to a model's list of different parameter sets for a particular node or link type. Many elements in a model can reference or "share" the same parameter set in a model's list. For example, many message sources could reference the same transport protocol in the model's list of transport protocols.

Each model list has a counterpart list of objects in the system library. As with the node and link parameter sets, each model list is initially populated with copies of the object from the counterpart list in the system library.

## 4. Modeling Constructs

### ***Creating:***

Parameter sets may be created either in the system library or in the local model.

If a parameter set is created in the system library (Via the Archive/Objects menu option), then it must be copied into the model in order to be accessed by the model. This is done automatically when a new model is made, or manually later by using the Add button on the parameter set list box.

A parameter set may also be created locally in the model. A model's parameter sets can be accessed using Define/Node Parameters or Define/Link Parameters on the menubar. In addition, a model's parameter sets can be accessed from a node or link dialog box by clicking on the detail button next to the Parameters entry. Remember that these parameters sets can be shared among many nodes or links in the model, even if the parameter set is created from a particular node or link's dialog box.

The objects in the other kinds of model lists can also be accessed using Define on the menubar. In addition, a model list can be accessed from the dialog box for the object that needs to reference a member of the model list. The Add button is used to copy a library parameter set to a new local parameter set with a different name. The new parameter set may then be edited. Nodes or links in the model may then reference this parameter set.

There is no provision to move a locally created object into the system library. The parameter values must be retyped via the Archive/Objects menu option.

### ***Connectivity:***

A parameter set defined in the system library is available to all models.

A parameter set defined locally is only accessible in the model where it is defined.

### ***Editing:***

To edit a local definition pull up the parameter set list box, highlight the parameter set to change, and then select Edit. Any object which uses the parameter set will see the changes.

To edit a system library definition use the Archive/Objects menu option to select and edit the required parameter set. The changes are saved in the system library but are not automatically transmitted to existing models.

To gain access to changed system library parameter sets, the local copy of the parameter set must be deleted (via the Define menu option) and the updated system parameter set recopied from the system library via the Add button on the parameter set list box.

Any references to the local parameter set must be deleted before the parameter set can be removed and recopied from the system library.

### ***Execution:***

N/A

### ***Reporting:***

N/A

### ***Fields:***

*See discussion of parameters under each type of node or link or under each type of model list element.*

## 4.64 Penalty Tables

### *Purpose:*

If Minimum Penalty routing is selected then the route chosen between origin and destination is the one which has the least total penalty. Minimum Penalty routing may be specified independently for packet-switched and circuit-switched traffic.

“Penalty” is a weighting factor applied to a link. It may be used to represent distance (higher penalty = longer distance), cost (higher penalty = higher cost), or other metrics as desired by the user. Penalties are entered as integer values by routing class in tables. A particular table is then applied to a particular link to define the link penalties for the different routing classes using the link.

A negative penalty for a routing class forbids that routing class traffic from using that direction of that link.

Penalties may also vary depending on utilization of link bandwidth. It may be that increasing utilization causes increased congestion and so it is desired to route traffic away from congested links by increasing the penalty incurred for using the link.

### *Creating:*

The penalty values are entered as tables. There may be several tables in the same model. Once a table has been created it may then be applied to a specific link; the link picks up the penalties as defined in the table.

One table may be applied to several links. Consequently the task of updating the penalty applied to several links may result in only one table requiring update.

Separate tables for packet-switched traffic and circuit-switched traffic are maintained.

### *Connectivity:*

The penalty tables are applied to arcs connecting the nodes to the links, and they represent the penalties for traffic entering the link from that node. The penalty in one direction over the link may be different from the penalty in another direction by applying different tables at either end of the link. Different routing classes may apply different penalties to the same link via multiple columns in the table.

Packet-switched traffic may route over all types of link. Each port connection between a node and a link may have a penalty table attached to it which defines the penalty for using that port for output to the link.

Penalty tables are global in scope—they may be seen by all links in the model including those in subnetworks and in the backbone.

Note that a routing algorithm is chosen for the backbone and individually for each subnetwork. If Minimum Penalty based routing is selected it will apply for the specific backbone or subnet. The routing decision being made is then to reach the destination in the backbone or subnet, or to reach an access point that gets closer to the destination, with the least incurred penalty.

### *Editing:*

The Penalty Tables are accessed from the **Define/Routing Penalties** menu option. A list of tables is presented. Select the one to edit, or add a new one. A default penalty table is always initialized in a model with the name “One Hop,” which is also set by default to each new arc in the model. This table assigns a

#### 4. Modeling Constructs

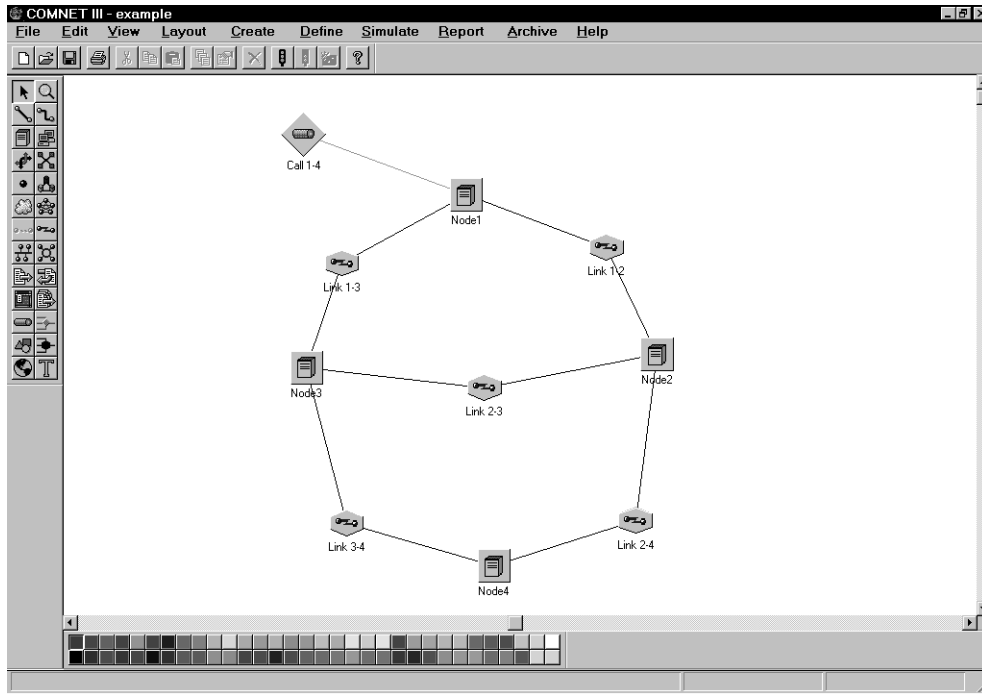
penalty of 1 for all routing classes and if it is the only table used in the model then the routing algorithm reverts to minimum hop.

As an example, take the network shown below. The primary route from Node1 to Node4 is over link 1-3, then link 3-4. In order to achieve this by the use of penalty tables, two tables are required.

The “Alternate Route” table is defined to apply a cost of 3 or more to each link. This cost is increased with increasing link utilization.

A new table called “Best Route” is created which has a default penalty entry of 1. This is applied to link 1-3 and link 3-4.





Example Network

Call Routing Penalty Table

Name:

OK  
Cancel

Edit Entry:

Threshold	Default	Standard		
0.000000	3			
30.000000	<input type="text" value="5"/>			
50.000000	10			

Insert Row  
Delete Row

Alternate Route Routing Table

## 4. Modeling Constructs

Threshold	Default	Standard		
0.000000	1			

**Best Route Routing Table**

To apply the table to a link, double click on the arc which connects the link to the routing node. The arc dialog box then enables the penalty table to be specified.

**Execution:** *See Routing Protocol: Call*  
*See Routing Protocol: Packet*

**Reporting:** N/A

### **Fields:**

**Name** The name of the table.

**Threshold** One row may be entered per threshold level to make the table adapt to different utilization levels on the link. Note that the first Threshold value will always be 0; despite the fact that it is editable, you should not need to change it.

The first row is from 0% up to the next threshold level (30% say). The next row to the next level (to 50% say). The last row is from whatever the last value is through to 100%. The specific threshold values required are entered by the user. As many rows as desired may be added.

For packet-routing penalties, the threshold values can be utilization percentage or delay (in seconds), depending on the congestion threshold type chosen on the penalty routing dialog box. *See Routing Class: Packet.*

If only one row is present it covers the range 0% to 100%.

**Penalty Per Routing Class** Across each row are a number of columns, one column per routing class. The actual penalty value for the routing class at the threshold level is entered here.

The first column is titled "Default." This value is applied to all routing class columns across the row, unless an actual value is entered in a specific routing class column. This means that if all routing classes across the row need the same penalty value, it need only be entered once. Or only those values that differ from the default need be entered.

There is generally a column for the “Standard” routing class. When a new model is made, COMNET III introduces a single routing class called “Standard” which appears as the available choice on the message and call generator dialog boxes. The “Standard” routing class is just like any other user defined routing class and so appears as a penalty table column, unless it is deleted or renamed by the user.

## 4. Modeling Constructs

### 4.65 Protocol Rate Controls

*Purpose:*

The rate control options for protocols model those traffic sources that generate traffic at a rate slower than the line speed available or for modeling those algorithms that regulate the admission of traffic on a network to respond to some traffic burst constraint to network congestion. Examples of traffic that may make packets available at slow rates include constant rate sources such as telephone, video, or sensor outputs, and traffic that may be more variable are compressed versions of the constant rate sources as well as more bursty sensing. Examples of the intentional regulation of traffic are the rate throttling at access to frame relay services in the presence of congestion or the available-rate algorithm for ATM.

All of the rate controls involve burst measurements that measure all the traffic from a particular source (or traffic command in a particular application source or connection through a transit net) to a particular destination. A source may send multiple messages to the same destination so that these messages overlap and all these message instances will be governed by the same rate control. This behavior allows using the rate controls for modeling services such as ATM and frame relay which may aggregate many instances on the same permanent virtual circuit (PVC) or usage parameter control (UPC) function.

## 4.66 Protocol Rate Controls: Available Rate

*Purpose:*

The available rate algorithm responds to network congestion like the throttled rate does, but according to the available bit rate (ABR) algorithm developed by the ATM Forum. The algorithm has been generalized somewhat to handle different ways to measure bursts but it is the ATM ABR algorithm when the burst is measured with packets and the packets are fixed in size at 53 bytes total.

The screenshot shows a dialog box titled "Available Rate Control (ATM TMS 4.0)". It contains the following fields and values:

- Name: DEFAULT
- Burst units: Packets
- Nrm: 32
- PCR: 100.000000
- MCR: 5.000000
- Limit: 100
- ICR: 25.000000
- CIF: 10
- Available rate computation:
  - AIR: 5.000
  - RDF: 2.000000
- Reduction for insufficient activity:
  - TOF: 4.000000
  - TDF: 2.000000
- Resource Management Packets:
  - RM Packet Priority: 1
  - RM Pkt Size (Bytes): 53
  - RM Pkt Protocol ID: (empty dropdown)
- Reduction for lost or late RM pkts:
  - Xrm: 16
  - XDF: 2.000000

Buttons: OK, Cancel

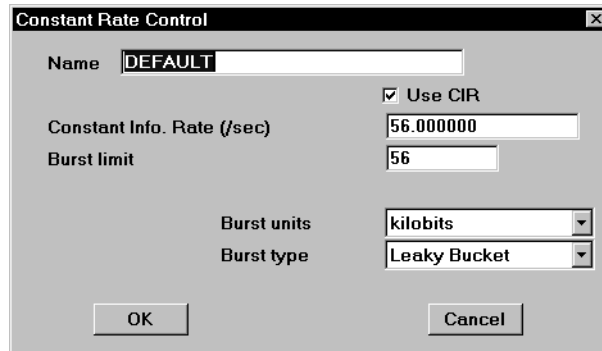
The algorithm involves sampling the congestion in the network with a resource management (RM) packet. The RM packet goes to the destination which then returns it to the source. If any intermediate node or the destination is congested when the RM packet comes by, the RM packet gets its congestion flag set. When the RM packet arrives back at the source, the source is allowed to additively increment the available rate if the RM does not have its congestion flag set, or multiplicatively reduce the available rate if the RM packet indicates congestion in the network.

## 4. Modeling Constructs

### 4.67 Protocol Rate Controls: Constant Rate

*Purpose:*

The constant rate algorithm allows packets to be created as long as the addition of the packet to the burst doesn't exceed the burst limit for the connection. If the packet would exceed the burst limit, the algorithm delays its creation until there is room available in the burst. The burst may be measured with one of three burst measurement algorithms (leaky bucket, sliding window, jumping window) and may be based on the kilobits transmitted or the number of packets transmitted.



The image shows a dialog box titled "Constant Rate Control". It contains the following fields and controls:

- Name:
- Use CIR
- Constant Info. Rate (/sec):
- Burst limit:
- Burst units:
- Burst type:
- OK button
- Cancel button

If the burst size is in packets and the limit is one packet, then the constant rate algorithms will transmit packets periodically, one packet at a time. With larger burst limits, the packets will be allowed to be transmitted as fast as possible but only up to the limit and then the source must be idle for a period of time so that the average rate is the specified value. The dynamics of how many packets are transmitted and how long the idle interval is depends on the type of burst algorithm is used.

A constant rate algorithm with large burst limits also can model bursts of packets while maintaining some average data rate.

## 4.68 Protocol Rate Controls: Throttled Rate

*Purpose:*

The throttled rate control normally allows traffic to be unconstrained and may be allowed to introduce packets into the network as quickly as possible. When congestion is indicated in the network by the presence of backward or forward explicit congestion notification (BECN or FECN), a constant rate control is activated so that the rate is reduced to that constant rate on average. After receiving indication that the congestion is no longer in the network, the control is deactivated and the rate returns to unrestricted again.

**Throttled Rate Control**

Name:

**Congestion Detection**

Congestion Direction:

**Recovery Detection**

Min. Congestion Count:

Min. Recovery Count:

**Rate Limit During Congestion**

Timeout delay (sec):

Use CIR

Constant Info. Rate (/sec):

Burst limit:

Burst units:

Burst type:

OK Cancel

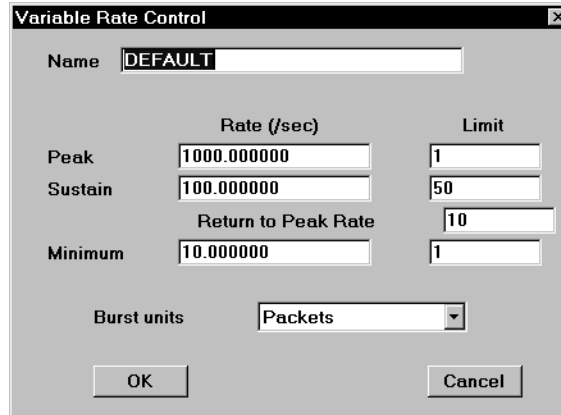
This control particularly addresses reducing PVC data rates to the committed information rate (CIR) in the presence of network congestion for frame-relay routers and frame-relay access devices (FRADs).

## 4. Modeling Constructs

### 4.69 Protocol Rate Controls: Variable Rate

*Purpose:*

Variable rate control is similar to the constant rate. In the constant rate control, the packets are either created as quickly as possible when the burst is available or not at all when the burst is not. The variable rate control allows the peak rate to be controlled (when the burst is available) and allows a minimum rate to generate a few packets when the burst is unavailable. The rate toggles between peak rate and minimum rate so that a sustainable rate is achieved on average according to the rate control.



The image shows a dialog box titled "Variable Rate Control". It contains several input fields and a dropdown menu. The "Name" field is set to "DEFAULT". There are three rows of rate and limit settings: "Peak" with a rate of 1000.000000 and a limit of 1; "Sustain" with a rate of 100.000000 and a limit of 50; and "Minimum" with a rate of 10.000000 and a limit of 1. A "Return to Peak Rate" field is set to 10. The "Burst units" dropdown menu is set to "Packets". There are "OK" and "Cancel" buttons at the bottom.

	Rate (/sec)	Limit
Peak	1000.000000	1
Sustain	100.000000	50
Return to Peak Rate		10
Minimum	10.000000	1

Burst units: Packets

This algorithm involves two burst measurements: one for the overall sustainable rate and the other for the current peak or minimum rate. The bursts are measured with the leaky bucket algorithm; the other options are not available because they are less efficient and this is a more complex rate control.



## 4.70 Received Message Scheduling

### *Purpose:*

When a message is sent, there is a text label associated with the message. This is a mechanism used to trigger desired sources at the destination once the message has arrived; it is usually not a description of the message or its contents. This is particularly used when received message scheduling is defined for applications, responses etc. Received Message Scheduling means that when a message with the required message text arrives at the destination node then the scheduling requirement for some new generator at the destination node has been met and may be started.

For instance, a model of a client server architecture may be built. Each client may have a message generator which sends a query message to the server with the message text "Database Query." On the server there may be an application which is scheduled by received message and is looking for the message text "Database Query" to be received. When a message with this text is received, a new instance of the server application may be started.

Because the message text is separate from the message name, a number of different message sources may send the same message text.

### *Creating:*

Message Text is defined on the dialog box which describes the message or transport command.

### *Connectivity:*

Messages are generated by Message Sources, Response Sources, Session Sources, Transport Commands, Setup Commands and Answer Commands. Wherever a message may be generated, a message text may be entered.

### *Editing:*

Via the appropriate message generator dialog box.

### *Execution:*

When a message is created, its message text is set based upon the message text settings. This is either a copy of the message name, the text of the original message, or specific text entered by the user. This text label is then associated with the message as it moves to its destination.

When the message arrives at the destination its text may be used to schedule some activity at the destination, such as a message source or a response source or an application.

If the message text is not required by any source at the destination the message and its associated text are discarded.

If the text is specified as a requirement by some source at the destination then the message text is placed in a received message list at the destination and remains there until such time as it is used to satisfy a scheduling requirement. A message text from a particular message instance will only satisfy one scheduling requirement at the destination, after which the message text is considered used and so discarded.

It may be that at the destination for the message there are several traffic sources or applications which require the message text before they can be scheduled. The receiving node keeps an application prototype list, which is a list of all the applications (including traffic sources and response sources) which may run on the node. When a message is received this list is scanned in order to determine

## 4. Modeling Constructs

whether the received message satisfies any message requirement. If so, a new instance of the application is scheduled and the message is considered as having been used. If an application has multiple message text requirements, the application instance is not created until all of its requirements are satisfied. The prototype which “consumed” the message is then moved to the end of the prototype list so that it will be considered last next time.

This means that if two entries in the prototype list have the same message requirement they will be satisfied alternately as the particular message arrives, assuming each has only one message text requirement.

It may be that one or more entries in the prototype list require several messages texts before a new instance may be created. When one message arrives no message requirement is completely satisfied. In this case, the message is saved in the received message list. When the next message arrives there are now two messages which could possibly satisfy a message requirement. The application prototype list is scanned in order—as applications within the prototype list are found that can have all of their message requirement satisfied a new instance of the application is started, the messages used, and the prototype entry moved to the end of the prototype list.

If Application1 requires MsgA and Application2 requires MsgA and MsgB the following may happen. A MsgA arrives which satisfies Application1. A new Application1 is created, the MsgA used, and Application1 moved to the end of the prototype list. Another MsgA arrives. Application2 is tested as it is at the top of the prototype list but it cannot be started as there is no MsgB. Application1 is tested and its requirement satisfied—another Application1 is started. The only way an Application2 can start is for a MsgB to arrive first. The MsgB cannot start Application2 on its own so MsgB is saved in the received message list at the node. When a MsgA arrives there is now a MsgB already there. The prototype list is scanned and, as Application2 is now at the top, the MsgA and Msg2 requirement for Application2 can be met. Therefore a new Application2 is started, the MsgA and MsgB “consumed,” and the Application2 prototype put to the end of the prototype list.

The message delays included in the output reports are based on the creation time of the message text for a message. For a message that has its text set based upon the original message, the message delay is measured from the creation time of the incoming message text. This provides a mechanism for measuring round-trip delay when an incoming message triggers a response and the response message uses the original (i.e., incoming) message’s text.

**Reporting:**

N/A

**Fields:**

**Delay**

Application message, response, and session sources that use received message scheduling can now include a delay measured from when all message requirements satisfied until the application attempts to begin execution. No processor is required during this delay, thus multiple source instances may be delaying simultaneously after being triggered by received message.

**Message Text**

From the sender-of-the-message point of view, the text of the message must be

set. The options for setting the text are:

### **Use Original Message**

Use the text of the message which scheduled the application, response, etc., which is creating the new message.

### **Copy Message Name**

Use the name of the message as the message text.

### **Set Message Text**

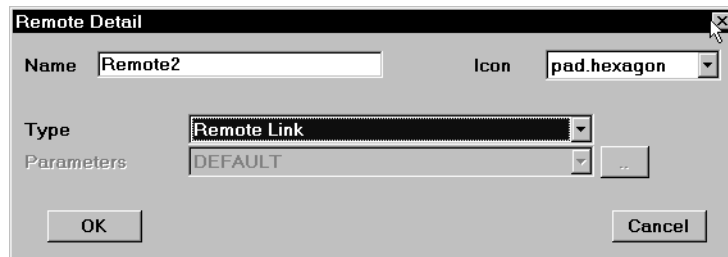
Explicitly set the message text with a label typed by the user.

## 4. Modeling Constructs

### 4.71 Remotes

#### *Purpose*

The remote icon is used to allow clustering of arcs without having to draw those arcs all the way to the main icon. An example is a multiaccess link connected to a large number of nodes: the remote link can allow several nodes to be clustered and connected to the remote link so that there is just one arc from the remote link at each cluster to the multiaccess link icon. Alternatively, a router may connect a group of links that are located far from the router in the graphical layout. In this case, a remote node icon can connect to each link in the group close to where the links are drawn and there only needs to be one arc drawn from the remote node to the router. Remote nodes and links have no modeling significance--they serve only to reduce graphical clutter.



## 4.72 Reports

### *Purpose:*

To report on the performance of the modeled network in a tabular fashion. The reports produced are written out to file in an 80 column ASCII format with ASCII page breaks for 8 1/2" x 11" paper and a fixed 10 characters per inch font.

The tabular reports give a statistical assessment of performance. Generally, mean values, maxima, minima and standard deviations are reported. These include delay times, response times, device loading and utilization, etc. These reports provide information useful for judging performance of the model.

### *Creating:*

A set of reports is produced automatically at the end of each replication. For the first replication they are saved in the file **Stat1.rpt**. Each subsequent run of the model will replace the contents of the **Stat1.rpt** with the newly collected results. If you wish to save the contents of a simulation run to a different report you will need to set a new file name for the report by selecting the Report Menu option and choosing the Set File Name option. Click on the **Add** button and type in a new report name **Stat2.rpt**. Then click on the **Done** button and when the simulation is rerun the results will be saved to the newly named report file. These files are located in a directory which has the same name as the model.

Also, restarting a simulation automatically deletes reports from previous runs. Beware of this and if necessary rename or move desired output files from earlier runs so they will not be deleted.

Saving a model to a different name will automatically create a new output directory and thus protect old reports.

### *Connectivity:*

N/A

### *Editing:*

To review the results and to print them out use the Simulate/Browse Reports menu option. The dialog prompts you to enter the replication you want to review. The respective reports are then shown in a separate window using your system text editor.

You may also open the report text files in a word processing package. For best results, open the file as ASCII text (no formatting) and use a fixed-space font (such as Courier) small enough to fit the full record on each line.

A different text editor can be used by changing the comnet.ini file found in the COMNET III executable directory (not to be confused with the Windows system.ini file on the PC implementation).

### *Execution:*

As the simulation executes statistics are collected in memory. When the replication ends the collected statistics are used to write out the reports.

*If a model is re-run then the reports from previous runs are overwritten.*

### *Reporting:*

N/A

### *Fields:*

### **Reports**

On the Report menu option you may select which reports to collect for a particu-

#### 4. Modeling Constructs

lar model being worked with. This selection is saved as part of the model data file. See the reports section of the manual for a fuller description of each report.

##### **Show Group Node Detail**

A Group Node is the same as a Computer & Communications node except that it has a quantity field. This specifies how many identical copies of the node are to be included in the simulation when it runs—but the copies do not appear on the user interface.

From a reporting point of view, with “Show Group Node Detail” off, only totals for all nodes in the group are reported. If it is set to on, then a line appears for each node in the group on each report, followed by a total for all nodes in the group.

##### **On**

When checked, this turns the selected report(s) on for the simulation run.

##### **Select All**

Selects all reports in the list.

## 4.73 Routing Class: Call

### *Purpose:*

Routing Class is a label which is applied to a call source. It may be thought of as a description of the type of call which applies various characteristics and parameter values to the call.

Different Routing Class labels may be applied to different call sources. As many Routing Classes as desired may be defined in the model.

When either the Minimum Penalty or the User-Defined Table routing protocols are used, different penalties or different routes may be defined by Routing Class. This allows different “types” of traffic to be routed separately over the same network. For instance, routing classes for “Voice Traffic” and “Fax Traffic” may be specified. Selection of penalty table or user-defined table for routing can then be made based on whether the traffic is for voice or for fax.

Call routing classes are also important for setting bandwidth requirements for call sources.

Note that each routing class has associated with it a complete routing table. Keep the number of classes to a minimum in order to reduce memory requirements for the model and to reduce processing time for updating tables.

### *Creating:*

To create a Routing Class edit the appropriate type of traffic source (call source in this case) and click on the double dot button at the side of the routing class box. Also the **Define/Routing Classes** menu option may be used to create and/or edit routing class definitions.

A Routing Class definition may also be added to the system library via the **Archive/Objects** menu option. If a routing class is defined in the system library it will become available when a new model is made, or when the system library definition is specifically copied into the local model.

COMNET III always creates a routing class called “Standard” when a new model is made.

### *Connectivity:*

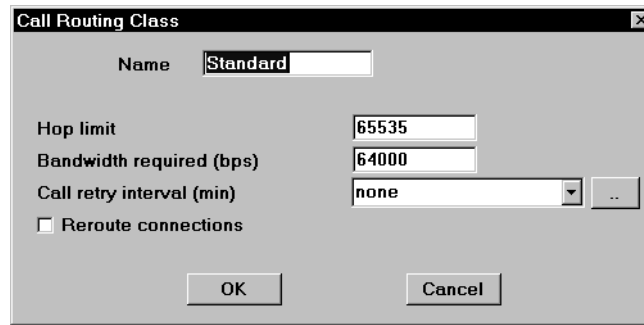
A call source references a routing class label in the model’s list of call routing classes.

### *Editing:*

Double click on a call source icon to bring up the call source dialog box, or highlight the icon and pick the **Edit/Detail** menu option. The Routing Class to use for the call source can then be applied using the routing class pulldown box.

The Routing Class parameters themselves are edited by clicking on the double dots next to the routing class pulldown box, and then by selecting the **Edit** button.

## 4. Modeling Constructs



The image shows a dialog box titled "Call Routing Class". It contains the following fields and controls:

- Name:** A text box containing the word "Standard".
- Hop limit:** A text box containing the number "65535".
- Bandwidth required (bps):** A text box containing the number "64000".
- Call retry interval (min):** A dropdown menu currently showing "none", with a small "..." button to its right.
- Reroute connections:** A checkbox that is currently unchecked.
- Buttons:** "OK" and "Cancel" buttons at the bottom.

**Call Routing Class Dialog Box**

**Execution:** See field descriptions below.

**Reporting:** N/A

### **Fields:**

**Name** The name of the routing class.

**Hop Limit** When a call is created an attempt is made to route it using the routing protocol. The Hop Limit places an upper bound on the number of hops that may be used on a selected route before the call is blocked on that route. The Hop Limit applies to all routing protocols. Different routing classes may have different Hop Limit values.

**Bandwidth Required** When a call is created and routed across a network, the call is a requirement to acquire and hold bandwidth for a period of time between origin and destination.

The amount of bandwidth to request and hold is specified as the Bandwidth Required on the call routing class.

For instance, ISDN carries voice with a 64 kbps requirement.

Although specified in kbps, the bandwidth parameter may be interpreted as analog kilohertz as long as that usage is consistent in a model. For example, an analog radio channel for voice may be 12 kHz

Remember that for circuit-switched traffic, the calls are carried over the circuit-switched capacity allocation on point to point links. If, when a call attempts to route over a link, the link has less free bandwidth than demanded by the Bandwidth Required for the call, then the call will be blocked from that link. A call can also be blocked at a node if the call's bandwidth requirement exceeds the node's available bandwidth.

**Call Retry Interval** If a call is blocked, the Call Retry Interval determines whether the call will be reattempted some time later.

If this interval is set to "None," then no reattempts will be made.



Note that a call may block on a primary route, but still be able to be carried on an alternate route. This will not invoke the Call Retry Interval. Only when a call has blocked on all routes, and cannot preempt on any route, would it inspect whether a Call Retry Interval were specified or not.

#### **Reroute Connections**

Calls are carried between origin and destination over intervening nodes and links. If a node or link on the route fails then a decision has to be made whether to re-establish the connection or not. If “Reroute Connections” is on, then, at a failure event, the routing tables are recalculated and a new attempt made to route the call. This will result in a different route being picked to carry the call. If Reroute Connections is off, the routing tables are still recalculated (for later calls), but no attempt is made to reconnect current calls.

## 4. Modeling Constructs

### 4.74 Routing Class: Packet

***Purpose:***

Routing Class is a label which is applied to a message source. It may be thought of as a description of the type of message which applies various characteristics and parameter values to the message.

Different Routing Class labels may be applied to different message sources. As many Routing Classes as desired may be defined in the model.

The Routing Class is also important for messages when either the Minimum Penalty or the User-Defined Table routing protocols are used. In this case different penalties or different routes may be defined by Routing Class. This allows different “types” of traffic to be routed separately over the same network. For instance, routing classes for “File Transfer” and “E-Mail” may be specified. Selection of penalty table or user-defined table routing can then be made based on whether the traffic is File Transfer or E-Mail.

Note that each routing class has associated with it a complete routing table. Keep the number of classes to a minimum in order to reduce memory requirements for the model and to reduce processing time for updating tables.

***Creating:***

To create a Routing Class edit the appropriate type of traffic source (message, response, session source, or equivalent command) and click on the double dot button at the side of the routing class box. Also the Define/Routing Classes menu option may be used to create and/or edit routing class definitions.

A Routing Class definition may also be added to the system library via the Archive/Objects menu option. If a routing class is defined in the system library it will become available when a new model is made, or when the system library definition is specifically copied into the local model.

COMNET III always creates a routing class called “Standard” when a new model is made.

***Connectivity:***

A message, response, and session source and a transport, setup, and answer command each have a reference to a routing class in the model’s list of packet-routing classes.

***Editing:***

Double click on a message source icon to bring up the source dialog box, or highlight the icon and pick the Edit/Detail menu option. The Routing Class to use for the source can then be applied using the routing class pulldown box.

The Routing Class parameters themselves are edited by clicking on the double dots next to the routing class pulldown box, and then by selecting the Edit button.

**Packet Routing Class Dialog Box**

**Execution:** See field descriptions below.

**Reporting:** N/A

**Fields:**

**Name** The name of the routing class.

**Hop Limit** When a message is created an attempt is made to route it using the routing protocol. The Hop Limit places an upper bound on the number of hops that may be used on a selected route before the message is blocked on that route. The Hop Limit applies to all routing protocols. Different routing classes may have different Hop Limit values.

**Session Retry Interval** If a session setup packet is blocked, the Session Retry Interval determines how long before the session is retried.

If this time is set to “None,” then no reattempts will be made.

Note that the transport protocol determines whether data packets are retransmitted or not. The Session Retry Interval only determines the behavior of the session setup packet.

**Reroute Connections** Sessions are carried between origin and destination over intervening nodes and links. If connection oriented routing is used then all packets follow the same virtual route. If connectionless routing is used then packets are individually routed.

If connection-oriented routing is in use and if a node or link on the route fails, then a decision has to be made whether to reestablish the connection or not. If “Reroute Connections” is on, then, at a failure event, the routing tables are recalculated and a new attempt made to route the session. This will result in a different route being picked to carry the session. If “Reroute Connections” is off, the routing tables are still recalculated (for later sessions), but no attempt is made to

#### 4. Modeling Constructs

reconnect current sessions.

If connectionless routing is being used, then upon node or link failure, only the packets at the node or link need to be retransmitted according to the transport protocol settings.

##### **IGRP Routing Bandwidth Factor & IGRP Routing Delay Factor**

If messages which use the routing class are being routed via an IGRP routing algorithm, a bandwidth, utilization, and delay factor are required for the routing algorithm to calculate the metric for using a particular link. The default values are the standard IGRP default values

*See “Routing Protocol: Packet” for more detail.*

## 4.75 Routing Protocol: Call

### *Purpose:*

To make a routing decision for a call that has not reached its destination.

As call traffic arises in COMNET III, a route must be picked via a sequence of nodes and links in order for the call to reach the destination.

In a real network this function may be accomplished in many ways. Examples include by a centralized controller, by distributed tables maintained in every switch, or by separate packet-switched routing transactions across the network. COMNET III does not mimic exactly the logic and sequencing of such decisions. Rather, the goal is to model the effect of the real routing mechanism and to provide in the simulation a routing protocol which arrives at the same routing result.

For instance, the end result of routing decisions in many networks is to establish the least hop route across the network. In COMNET III there is a Minimum Hop routing protocol. This arrives at a least hop route between an origin and destination, and so exhibits the same result as a real network using minimum hop routing.

When a link or node fails, in COMNET III the minimum hop tables which are kept internally are instantaneously (in simulation time) recalculated and are then available at the same instant for all calls which require a routing decision. This is not the same as a real network, but the COMNET III approach is valid when used for overall network capacity planning. If the goal of your simulation is to establish the detailed sequencing of routing table updates in the case of a failure event, the standard COMNET III simulation environment does not capture sufficient detail.

### *Creating:*

The routing protocols are built into COMNET III. For call traffic, the protocols which are provided are:

- Minimum Hop
- Minimum Penalty
- User Defined Routing Tables

If minimum penalty routing is used, then penalty tables will need to be created and applied to link arcs. If User-Defined Table routing is used then routing tables must be manually created.

### *Connectivity:*

A routing protocol is applied to the backbone network, and independently to each subnet. If a call originates in one subnet and has a destination in another subnet then 3 routing protocols would be queried—1 for each subnet and 1 for the backbone.

The origin and destination may be in the same subnetwork or backbone, or in different subnetworks. The possibilities are:

<u>Origin Network</u>	<u>Destination Network</u>	<u>Number of Routing Algorithms Applied</u>
Backbone	Backbone	1
Backbone	Subnet	2
Subnet	Same Subnet	1
Subnet	Backbone	2

## 4. Modeling Constructs

Subnet                      Different Subnet                      3

If a route crosses a boundary between a subnet and the backbone then an access point must be involved. In the backbone, the backbone routing protocol will route the call to/from an access point. In the subnet the subnet routing protocol will route to/from the closest access point.

### *Editing:*

The routing protocol selected for the backbone or particular subnet may be changed at any time. The backbone routing protocol is specified on the **Define/Backbone Routing** menu option.

The subnet routing protocol is specified by highlighting the subnet icon and the selecting the **Edit/Detail** menu option. (Double clicking on a subnet icon causes the subnet to be entered.)

### *Execution:*

#### **Minimum Hop Routing**

The Minimum Hop Routing protocol finds the least number of hops between the origin and destination.

When the simulation starts each node calculates all the possible routes to the other nodes in the model. A table is stored at each node which lists the next link/next node pair to use for each route to reach a particular destination. The link/next node pair entries in the table are ordered by increasing hop count of the route to which they belong. The link at the top of the table belongs to the route with the least hop count and the route at the bottom belongs to the route with the largest hop count. Links for routes with the same hop count are arbitrarily ordered.

When a call requires routing, the link/next node pair at the top of the table is inspected. This is the link that follows the least hop route to the destination. The link must pass the following tests:

- be inside the hop limit for the routing class
- have sufficient bandwidth available
- the node at the other end of the link must have sufficient bandwidth
- not connect to a previously visited node

If the link passes these tests then the call may use the link (and the next node). The routing decision then passes to the next node and an outgoing link found. By repeating this process the route is built up link by link and the required bandwidth acquired until the destination is reached. When the destination is reached, the bandwidth will be held for the duration of the call. Note that this routing process is instantaneous in simulation time.

If the first link/next node pair in the table at a routing node does not pass these tests the next pair is checked, and so on until an outgoing link is found.

If no route is found, and if the preemption flag is set to yes, then the routing protocol will start back at the first node and reattempt to route the call by testing node by node whether sufficient capacity can be preempted. If a complete route can be found with preemption, then existing call(s) will be preempted and the call routed.

If no route can be found either normally or by preemption, then the call will be blocked and optionally retried later depending on the routing class characteristics.

If there is more than one route from a node with the same hop count, then the ordering of the link/next node pairs for these routes in the routing table is arbitrary. One of them will be at the top and so will be preferentially picked by the routing protocol. When this pair fails a routing test (such as having insufficient bandwidth for a particular call), the next pair in the table for another route which has the same hop count would come into play. This pair is then moved above the other pair in the table and so becomes the preferred link/next node pair until it in turn becomes full.

Rather than waiting for links to go full before other links on routes of the same hop count are considered, a “Deviation Percent” parameter is provided. If this has the value 0 then the operation is as described above. If the deviation percent is positive then links for routes that are within the deviation percent of the shortest route will be considered equivalent and routed over on a round robin basis. For instance, a route of 4 hops and a route of 5 hops are considered the same length if the deviation percent is 25% or more.

The destination of the route may be an access point at a subnet/backbone boundary. The routing protocol establishes that a route exists as far as this point and so the bandwidth on each node and link comprising the route is claimed. Another routing decision may be required to carry the call further. If this decision succeeds then the call is carried. If this decision fails to find an onwards route then the bandwidth claimed from the earlier decision must be given back. Note that this bandwidth would have been instantaneously relinquished as the routing decisions are instantaneous in simulation time.

At the point of link or node failure the minimum hop tables are recalculated. Routes which contain failed components are no longer valid and so the link/next node pair references are removed from the node routing tables. When the node or link recovers, routes are again recalculated and link/next node pair references placed back in the node routing tables.

If a routing decision is being made at some intermediate node, it may be that the first choice link/next node pair which follows the minimum hop path to the destination has insufficient capacity to carry the call. In this case the next pair will be tested, and so on, until a pair is found or the call counted as blocked. Assuming a pair is found, this pair will reflect the minimum hop route that is available at this instant between the currently routing node and the destination. However, this route may not be the minimum hop route between the origin of the call and the destination—some better route may exist if a different routing decision were taken earlier than the current routing node. COMNET III does not “crank back” from the current node to earlier nodes under minimum hop routing to find better routes. However, it does provide a routing update interval which forces the routing tables to be recalculated periodically. This means that at the update time, a complete end-to-end analysis is performed to establish the best minimum hop routes. Consequently, you may observe nonoptimal routes being picked between a time when a node or link becomes saturated and the next routing update time. To minimize the effects of this behavior the routing update interval should be set to a small value. However, frequent routing table updates will slow down the simulation.

## 4. Modeling Constructs

### Minimum Penalty Routing

The Minimum Penalty Routing protocol finds the route with the least total penalty between the origin and destination. Penalties are integer factors applied to links by the use of penalty tables. If the penalty for each link is 1 then Minimum Hop and Minimum Penalty routing are identical.

The penalty values can be made variable depending on link utilization. The norm would be that, as link utilization increases, the penalty on a link would increase and so cause the link not to be selected as part of a least penalty route. This is an adaptive routing strategy.

When the simulation starts each node calculates all the possible routes to the other nodes in the model. A table is stored at each node which lists the link/next node pair to use for each route to reach a particular destination. The link/next node entries in the table are ordered by increasing total penalty of the route to which they belong. The link/next node pair at the top of the table belongs to the route with the least total penalty and the route at the bottom belongs to the route with the largest total penalty. Links for routes with the same total penalty are arbitrarily ordered.

When a call requires routing, the link/next node pair at the top of the table is inspected. This is the hop that follows the least total penalty route to the destination. The hop must pass the following tests:

- be inside the hop limit for the routing class
- have sufficient bandwidth available
- the node at the other end of the link must have sufficient bandwidth
- not connect to a previously visited node.

If the link passes these tests then the call may use the link (and the next node). The routing decision then passes to the next node and an outgoing link found. By repeating this process the route is built up link by link and the required bandwidth acquired until the destination is reached. When the destination is reached, the bandwidth will be held for the duration of the call. Note that this routing process is instantaneous in simulation time.

If the first link/next node pair in the table at a routing node does not pass these tests the next pair is checked, and so on until an outgoing link is found.

If no route is found, and if the preemption flag is set to yes, then the routing protocol will start back at the first node and reattempt to route the call by testing node by node whether sufficient capacity can be preempted. If a complete route can be found with preemption, then existing call(s) will be preempted and the call routed.

If no route can be found either normally or by preemption, then the call will be blocked and optionally retried later depending on the routing class characteristics.

If there is more than one route from a node with the same total penalty, then the ordering of the link/next node pairs for these routes in the routing table is arbitrary. One of them will be at the top and so will be preferentially picked by the routing protocol. When this link fails a routing test (such as being full), the next pair for another route which has the same total penalty would come into play. This pair is then moved above the other pair in the table and so becomes the pre-



ferred link/next node pair until it in turn becomes full.

Rather than waiting for links to go full before other links on routes of the same total penalty are considered, a “Deviation Percent” parameter is provided. If this has the value 0 then the operation is as described above. If the deviation percent is positive then links for routes that are within the deviation percent of the shortest route will be considered equivalent and routed over on a round robin basis. For instance, a route of total penalty 4 and a route of total penalty 5 are considered the same length if the deviation percent is 25% or more.

The destination of the route may be an access point at a subnet/backbone boundary. The routing protocol establishes that a route exists as far as this point and so the bandwidth on each node and link comprising the route is claimed. Another routing decision may be required to carry the call further. If this decision succeeds then the call is carried. If this decision fails to find an onwards route then the bandwidth claimed from the earlier decision must be given back. Note that this bandwidth would have been instantaneously relinquished as the routing decisions are instantaneous in simulation time.

At the point of link or node failure the penalty tables are recalculated. Routes which contain failed components are no longer valid and so the link/next node pair references are removed from the node routing tables. When the node or link recovers, routes are again recalculated and link/next node references placed back in the node routing tables.

The purpose of the routing update interval for minimum hop routing is to optimize the routing tables as links or nodes become saturated

If a routing decision is being made at some intermediate node, it may be that the first choice link/next node pair which follows the least penalty path to the destination has insufficient capacity to carry the call. In this case the next pair will be tested, and so on, until a pair is found or the call counted as blocked. Assuming a pair is found, this will be the lowest penalty pair of those pairs that have capacity to carry the call. However, this may not reflect the minimum penalty route between the origin of the call and the destination—some better route may exist if a different routing decision were taken earlier than the current routing node. COMNET III does not “crank back” from the current node to earlier nodes under minimum penalty routing to find better routes. However, it does provide a routing update interval which forces the routing tables to be recalculated periodically. This means that at the update time, a complete end-to-end analysis is performed to establish the best minimum penalty routes with congested nodes and links in situ. Consequently, you may observe nonoptimal routes being picked between a time when a node or link becomes saturated and the next routing update time. To minimize the effects of this behavior the routing update interval should be set to a small value. However, frequent routing update intervals will slow down the animation.

The minimum penalty protocol allows the penalty on a link to vary depending on the link utilization. For example, the penalty may have the value 1 for 0-30% loading and 5 for over 30% loading. As the model runs and calls are routed over a link, the routing tables are recalculated every user defined time period. The utilization level on each link at the instant the routing tables are recalculated is then used to compute the minimum penalty routes.

## 4. Modeling Constructs

**User-Defined Routing Tables** The User-Defined Routing protocol selects a route from 1 or more routes that have been manually entered by the user. Multiple alternate routes may be entered. Partial routes may also be entered which route the call to some intermediate node. At the intermediate node one or more routes, must be available to forward the call towards the destination.

When a user enters a number of possible alternate routes a selection criterion is required to determine which alternate to use. These criteria include:

- Dynamic Alternate
- First Available
- Max Idle Bandwidth
- Random List
- Round Robin

A route in User-Defined routing is implemented by keeping a routing table at each node. Each table may contain a choice of single links to use to reach some next node, or it may list several links and so route the call all the way to the destination. These may be regarded as implementations of Node-By-Node routing and Source Node routing respectively. In addition, partial routes that only get part way to the destination are allowed.

The routing tables are entered for each node by bringing up the node dialog box and then selecting the Call Routing Table button. A Destination Selection dialog box then presents a table of destinations and routing classes. After highlighting a particular destination/routing class intersection, routes to that destination may be entered by selecting the “Edit Selected” button.

Destination	Standard		
Destination	1		
Subnet A.Node6	0		
Subnet B.Node6	0		
Subnet B.Node8	0		
Subnet A.Node8	0		

You have selected routes for Destination / Standard

**Destination Selection Dialog Box**

A Route Definition dialog box is then presented. This allows a number of routes to the particular destination/routing class combination to be entered. The routes may completely reach the destination, or partially reach the destination.

### Route Definition Dialog Box

With User-Defined Table routing, both primary and secondary routes may be defined. When a routing decision is made, the primary routes are inspected and a route found. If no route can be found from the primary routes, then the secondary routes are evaluated. If still no route can be found then the call is counted as blocked (unless preemption is on and a preemptive route found).

The “Avoid Backtracking” check box has no impact on model execution. When a route is being defined, and with the box checked, links which return to a node already in the route will not appear in the “Possible Next Hops” list. With the box unchecked, all outgoing links from a node appear in the “Possible Next Hops” list.

Backtracking may be useful for certain routers or equipment that must handle the call even if the call must go to that device and backtrack to return to its path.

If there are several routes defined at a routing node, then when a call originates and requires routing some tie breaking mechanism is required. These include:

#### Dynamic Alternate

The intent is that the primary route selection would be “First Available” say, and the secondary route selection is “Dynamic Alternate.” When a routing decision is made the primary routes are inspected and the first available route returned. An available route is one that is inside the hop limit and has sufficient capacity on all the nodes and links along its path. If no route is available in the primaries then, with “Dynamic Alternate,” one of the secondary routes with available capacity is picked at random. Thereafter, when a call originates the primary routes are retested and, if full, the same secondary route is used. This continues until such time as the secondary route cannot carry the call (i.e., insufficient capacity). At

## 4. Modeling Constructs

this time, another random pick from those secondary routes with sufficient capacity is made and a new secondary route established.

If a call cannot be routed over either the primary or secondary routes, and if preemption is off or no preemptive route can be found, then the call would be treated as blocked and optionally retried later.

This is the alternate route selection rule used in British Telecom's dynamic alternate routing algorithm.

### **First Available**

The routes are evaluated in the same order as defined on the user interface. Each route is tested in turn to establish whether it is within the hop limit, that each link has sufficient bandwidth, that each node has sufficient bandwidth to route the call, and that no previous routing node is revisited. The first route to meet these criteria is selected, the bandwidth claimed, and the call routed.

### **Max Idle Bandwidth**

A route comprises one or more hops. On each route one of the hops will, at the instant that the routing decision is made, have the least amount of bandwidth on that route. If there are several routes then there will be one hop on each route that has the least available bandwidth. These "least bandwidth" hops are compared and the route selected which owns the hop which has the most bandwidth. In other words, find the worst hop on each route and take the route that has the best, worst hop.

The route found must be within the hop limit and have sufficient spare capacity to carry the call, and not revisit any previous routing nodes.

### **Random List**

The routing table lists one or more routes which lead to or reach the destination. A sublist is made of these routes which includes those that stay in the hop limit and have sufficient capacity on each node and link to carry the call, and do not revisit any previous routing nodes. A random pick is then made from the routes in the sublist.

### **Round Robin**

The routing table lists one or more routes which lead to or reach the destination. The routes are evaluated in the order of definition starting with the route immediately after the one which was last used. The first route which is within the hop limit and has sufficient capacity on each node and link and does not revisit any previous routing nodes is selected.

Once the call has been routed to the destination, the claimed bandwidth on each node and link is held for the duration of the call. At the end of the call the bandwidth on each node and link is simultaneously released.

If no route is found, and if the preemption flag is set to yes, then the routing protocol will start back at the first route and test whether sufficient nodes and links on the route can be preempted so that there is capacity across the entire route. If so then existing call(s) will be preempted and the call routed. Each route will be tested in turn until either a route is found or there are no more routes to test.

If no route can be found either normally or by preemption, then the call will be blocked and optionally retried later depending on the routing class characteristics.

The destination of the route may be an intermediate node partially completing the route to the destination, or an access point at a subnet/backbone boundary. At this point another routing decision may be required to carry the call further. If this decision succeeds, then the call is carried. If this decision fails to find an onwards route, then the bandwidth claimed from the earlier decision must be given back. Note that this bandwidth would be instantaneously relinquished as the routing decisions are instantaneous in simulation time.

**Reporting:** N/A

**Fields:**

**Link Utilization Update Interval**

Not used for call routing. All link utilization figures used in the call routing protocols are instantaneous measurements made at the time the figure is required.

The “Link Utilization Update Interval” is used in the routing protocols under the packet-switched model.

*See Routing Protocol: Packets*

**Minimum Hop Routing**

**Routing Update Interval**

Every routing update interval period, the minimum hop calculations are performed to find the minimum hop routes through the network. The tables kept at each node which record which next link/next node pair to use from the routing node are updated.

This updating is included because the availability of nodes and links to route calls is not static. As congestion increases then some nodes or links may become fully loaded and are not therefore available for carrying more traffic. When a routing node sees the link/next node pair is full, it picks the next best link/node from its local table. While this is the best local decision to make, it may result in an end-to-end route which is not the minimum hop between the end points. To overcome this problem, the complete end-to-end routes are recalculated periodically and so take into account such congestion effects.

Note that a node/link failure or recovery event causes an immediate recalculation of routing tables.

**Deviation Percentage**

Indicates those routes which are to be treated as equivalent from a distance point of view, as a percentage difference from the shortest route. For example, 4 hops and 5 hops are considered the same if the deviation percent is at least 25%. Routes that are the same distance within the deviation percent will be selected for routing on a round-robin basis.

A deviation percent of 0 suppresses the round robin routing even on routes which have the same hop count. To round robin on such routes use a small positive deviation percent (such as 1%).

## 4. Modeling Constructs

### Preemption

Flag to indicate whether higher priority calls are allowed to preempt lower priority calls.

### Minimum Penalty Routing

#### Routing Update Interval

Every routing update interval period the minimum penalty calculations are performed to find the least penalty routes through the network. The tables kept at each node which record which next link to use from the node are updated.

This updating is included because the availability of nodes and links to route calls is not static. As congestion increases then some nodes or links may become fully loaded and are not therefore available for carrying more traffic. When a routing node sees the next link/node is full it picks the next best link/node from its local table. While this is the best local decision to make, it may result in an end-to-end route which is not the minimum penalty between the end points. To overcome this problem, the complete end to end routes are recalculated periodically and so take into account such congestion effects.

Note that a node/link failure or recovery event causes an immediate recalculation of routing tables.

In addition, the penalty values defined for a link may vary be defined to vary by link utilization. At the point of routing table recalculation the penalty value to use for a link is established by looking up in the penalty table the penalty value to use given the current utilization level.

### Deviation Percentage

Indicates those routes which are to be treated as equivalent from a total penalty point of view as a percentage difference from the route with the lowest penalty. For example, total penalty 4 and total penalty 5 routes are considered the same if the deviation percent is at least 25%. Routes that are the same distance within the deviation percent will be selected for routing on a round robin basis.

A deviation percent of 0 suppresses the round robin routing even on routes which have the same total penalty. To round robin on such routes use a small positive deviation percent (such as 1%).

### Preemption

Flag to indicate whether higher priority calls are allowed to preempt lower priority calls.

### User Defined Routing Tables

#### Primary Route Selection

A number of routes at a node may be specified to reach a destination, or to reach a node closer to the destination. Primary routes are those routes which should be tested first to establish whether a route is available. Where several primary routes are specified some tie breaking criterion is needed to decide which route to use. The selection criteria are

- Dynamic Alternate
- First Available
- Max Idle Bandwidth
- Random List
- Round Robin

See the above section on *Execution* for a fuller explanation of each criterion.

**Secondary Route Selection**

If the primary routes do not yield a route, and if secondary routes are defined, then the secondary routes will be tested to see if a route can be established. Where several secondary routes are specified some tie breaking criterion is needed to decide which route to use. The selection criteria are

- Dynamic Alternate
- First Available
- Max Idle Bandwidth
- Random List
- Round Robin

See the above section on Execution for a fuller explanation of each criterion.

**Preemption**

Flag to indicate whether higher priority calls are allowed to preempt lower priority calls.

## 4. Modeling Constructs

### 4.76 Routing Protocol: Packet

*Purpose:*

To make a routing decision for a packet that has not reached its destination.

Inherently COMNET III routes packets on a datagram basis; each packet has individual routing decisions applied to it.

If “Connection-Oriented Routing For Sessions” is specified then for session traffic only, the session setup packets are specifically routed. The data packets follow the route established for the session setup packets. This would be the most common setting for modelling virtual circuit based operation. Other nonsession packets are still routed in a datagram sense.

As packet traffic arises in COMNET III, a route must be picked via a sequence of nodes and links in order for the packet to reach the destination.

In a real network this function may be accomplished in many ways. Examples include by a centralized controller, or by distributed tables maintained in every switch. COMNET III does not mimic exactly the logic and sequencing of such decisions. Rather, the goal is to model the effect of the real routing mechanism and to provide in the simulation a routing protocol which arrives at the same routing result.

For instance, the end result of routing decisions in many networks is to establish the least hop route across the network. In COMNET III there is a RIP Minimum Hop routing protocol. This arrives at a least hop route between an origin and destination, and so exhibits the same result as a real network using RIP.

When a link or node fails, in COMNET III the minimum hop tables which are kept internally are instantaneously (in simulation time) recalculated and are then available at the same instant for all calls which require a routing decision. This is not the same as a real network, but the COMNET III approach is valid when used for overall network capacity planning. If the goal of your simulation is to establish the detailed sequencing of routing table updates in the case of a failure event, the standard COMNET III simulation environment does not capture sufficient detail.

The management and routing of packet-switched traffic is modeled completely independently of the circuit-switched traffic. Packet-switched traffic has its own switching resource on each node, and its own transmission bandwidth on each link. These are not shared with the circuit-switched traffic. Also the packet routing protocol is entered separately from the call routing protocol and maintains independent routing tables.

*Creating:*

The routing protocols are built into COMNET III. For packet-switched traffic the protocols which are provided are:

- IGRP
- Link-State Shortest-Path First
- RIP Minimum Hop
- Minimum Penalty
- Shortest Delay
- User-Defined Routing Tables



For IGRP, Link State Shortest Path, & Minimum Penalty, routing penalty tables may need to be created and applied to link arcs.

If User-Defined Table routing is used, then routing tables must be manually created.

***Connectivity:***

A routing protocol is applied to the backbone network, and independently to each subnet. If a packet originates in one subnet and has a destination in another subnet then 3 routing protocols would be queried—1 for each subnet and 1 for the backbone.

The origin and destination may be in the same subnetwork or backbone, or in different subnetworks. The possibilities are:

Origin <u>Network</u>	Destination <u>Network</u>	Number of Routing <u>Algorithms Applied</u>
Backbone	Backbone	1
Backbone	Subnet	2
Subnet	Same Subnet	1
Subnet	Backbone	2
Subnet	Different Subnet	3

If a route crosses a boundary between a subnet and the backbone then an access point must be involved. In the backbone, the backbone routing protocol will route the packet to/from an access point. In the subnet the subnet routing protocol will route to/from the closest access point.

***Editing:***

The routing protocol selected for the backbone or particular subnet may be changed at any time. The backbone routing protocol is specified on the **Define/Backbone Routing** menu option.

The subnet routing protocol is specified by highlighting the subnet icon and the selecting the **Edit/Detail** menu option. (Note: Double clicking on a subnet icon causes the subnet to be entered.)

***Execution:***

When the simulation starts, each node calculates all the possible routes to the other destination nodes in its local subnet (or locally in the backbone).

For the purposes of this discussion, a hop is considered to be the combination of link plus node at the other end. For instance, when routing over a multiaccess link such as token ring, there may be two different nodes on the ring which could be routed through. These would be represented as two different hops in the routing table even though they both refer to the same link. Or there may be two different links connecting to the same next node.

The order of hops in the routing table at a node is determined by the metric being used to evaluate alternate routes. These metrics are discussed in the following sections. Some general comments which apply to all metrics are required. Note that User-Defined Tables are somewhat different from the other metrics.

A weighting factor is calculated for each of these routes so that the best “route” can be ascertained. For each route starting at a node, there is a first hop over which the packets would be sent. Each such hop is tagged with the weighting fac-

## 4. Modeling Constructs

tor that was calculated for the end-to-end route.

Note that there may be several first hops that have the same weighting factors. The ordering of such hops in the list is arbitrary.

When a packet requires routing, the hops are evaluated in the order that they appear in the table. The first hop would reflect the best route under the current routing metric assuming that the route is currently available (i.e., has no failed components). For a hop to be accepted the following tests must be passed:

- the hop limit for the routing class must not be exceeded.
- both the node and link on the hop must be up.
- the hop must not return to a previously visited node.
- if a session setup packet is being routed, the link and next node must be under its session limit.

If these tests are passed then an attempt will be made to route the packet onto the hop. This can only succeed if the port output buffer on the routing node has sufficient capacity to accept the packet, and the routing node has sufficient spare total output buffer capacity. Assuming there is buffer capacity, the packet is placed in the selected port output buffer and then is transmitted onwards by the link protocol to the next node. A similar decision is made at the next node, and so on, until the packet arrives at the destination.

If the routing node does not have sufficient buffer space on the port output buffer for the selected hop, the packet is blocked and optionally retried later from the origin depending on the transport protocol settings. (A packet blocked on a port output buffer on an ATM node will remain in a port input buffer at the ATM node.)

If the first hop does not pass the tests outlined above, then the next best available hop is selected, and so on until a hop selection is made. If no hop satisfies the tests, then the packet is blocked and optionally retried later from the origin depending on the transport protocol settings. Note that the buffer tests are made after the hop has been selected.

As the packet is switched through nodes, placed in port input and port output buffers, and transmitted over links, the packet consumes the respective type of resource. The process of moving the packet from origin to destination therefore extends over simulation time, but the individual routing decisions made at each intermediate node are instantaneous.

Where there are several hops listed in the routing table for routes of the same total weighting factor, generally only the route at the top of the list will be picked. It is an arbitrary choice which of the routes will be at the top. To enable a load balancing mechanism, a “Deviation Percent” parameter is provided. If this has the value 0 then the operation is as described above. If the deviation percent is positive then hops for routes that are within the deviation percent of the shortest route are considered equivalent and routed over on a round robin basis. For instance, if the routing metric is RIP Minimum Hop then a route of 4 hops and a route of 5 hops are considered the same length if the deviation percent is 25% or

more when load balancing is in effect. Once the top hop has been selected it will be moved after the hops with the same weighting factor. This means that routes with the same weighting factors will be selected on a round-robin basis.

The destination of a routing decision may be an access point at a subnet/backbone boundary. There is no guarantee that once a packet has reached an access point that there will be an ongoing route to reach the ultimate destination—link or node failures may have made the destination unreachable in the subnet. Assuming the access point is reachable for the first routing decision, COMNET III will route the packet to the access point. The packet would then block at the access point as the subnet routing protocol would not have any available routes to use to reach the destination.

At the time of link or node failure the routing tables are recalculated. The hops listed in the routing table for routes which contain failed components are marked as such and so will not pass the availability test. When the node or link recovers, routes are again recalculated and listed first hops marked as available. If a particular destination is unreachable (for instance, the only link to it has failed) then there are no routes from any node to the unreachable node. In this case, all hops listed in routing tables would be marked as unavailable and so packets would block at the origin and not part way to the destination.

## IGRP Metric

The IGRP metric calculates compound route weighting factors based bandwidth, utilization and delay metrics.

When a routing table update is made the compound penalty calculated for each link is given by the formula:

$$K1 * \text{bandwidth factor} + K2 * \text{bandwidth factor} / (256 - \text{load}) + K3 * \text{Delay Factor}$$

**K1, K2 & K3** are specified on the routing class dialog box and they can vary by routing class. For standard IGRP they have the values 1, 0 & 1 respectively.

$$\text{Bandwidth Factor} = 10^{10} / (\text{Bandwidth})$$

The Bandwidth for links is expressed in bits per second, computed automatically by COMNET III based on link parameters.

$$\text{Load} = \text{Utilization Percentage} * 255$$

$$\text{Delay Factor} = \text{topological delay in units of 10 microseconds}$$

For LANs this defaults to 100 (corresponding to 1000 microseconds or 1 millisecond). For other links it defaults to 2000 (or 20 ms). In COMNET III, the delay factor is set from the penalty tables and thus the values may be arbitrary.

The IGRP default delay values are used if the penalty assigned to a port has not been changed from its default value of 1. If a port has been assigned some penalty value other than 1, that value is used as the delay penalty in the IGRP metric. It is only necessary to edit the packet routing penalties if the default IGRP delay values are not satisfactory. This might be the case for a satellite link, where a delay of 2 seconds would correspond to a delay penalty of 200,000.

The load calculation is based on the most recent link utilization estimate at the time of the routing table update. The interval for updating link utilization esti-

## 4. Modeling Constructs

mates is specified on the dialog boxes for backbone and subnet detail. If K2 is zero, you may want to set the update interval to a large value to eliminate unnecessary updates. Similarly, if K2 is left at its default value of 0, you may want to set the routing table update interval to a large value to eliminate unnecessary routing table updates.

Under IGRP delay weighting factors are additive across a route, but bandwidth factors are only calculated for the least bandwidth link on the route.

The routing table calculation is performed at the start of simulation, and then on each node or link failure or recovery event. In addition periodic updates are required as the penalties applied to each link may vary over time due to changing utilization levels.

The time period between updates is given in the “Routing Update Interval” field. If link penalties are changing due to changing link utilization, the link utilization is averaged over the time specified by the “Link Utilization Update Interval.”

The Deviation Percent is based on total composite metric of a route.

### **Link-State Shortest-Path Metric (OSPF)**

The Link State Shortest Path metric calculates route weighting factors based on integer penalty values applied to links by the use of penalty tables. If the penalty for each link is 1 then Minimum Hop and Link State Shortest Path routing are identical.

Only the penalty values in the first line of the penalty table are used. In other words, the penalty values are fixed by the user at the start of the simulation.

The routing table calculation is performed at the start of simulation, and then on each node or link failure or recovery event. As the link penalties are fixed, periodic updates are not required.

The Deviation Percent is based on the total penalty metric of a route.

### **RIP Minimum Hop Metric**

The Minimum Hop Routing metric calculates route weighting factors based on the hop count between the origin and destination.

The routing table calculation is performed at the start of simulation, and then on each node or link failure or recovery event. Periodic updates are not required as the hop count between nodes is static, unless there are failures.

The Link Utilization Update Interval is not used by this metric.

The Deviation Percent is based on the hop count of a route.

### **Minimum Penalty Metric**

The Minimum Penalty Routing metric calculates route weighting factors based on integer penalty values applied to links by the use of penalty tables. If the penalty for each link is 1, then Minimum Hop and Minimum Penalty routing are identical.

The penalty values can be made variable depending on link utilization or packet delay across the link. The norm would be that, as link utilization or link delay

increases, the penalty on a link would increase and so cause the link not to be selected as part of a least penalty route.

The routing table calculation is performed at the start of simulation, and then on each node or link failure or recovery event. In addition periodic updates are required as the penalties applied to each link may vary over time due to changing utilization or delay levels.

The time period between updates is given in the “Routing Update Interval” field. The value of link utilization which is used to look up the applicable penalty value in the link penalty table is the last completed utilization measurement averaged over the time specified by the “Link Utilization Update Interval.” Link delays are averaged over the “Routing Update Interval.”

The Deviation Percent is based on the penalty metric of a route.

### Shortest Delay Metric

The Shortest Delay metric calculates route weighting factors based on packet delays seen on each link. Packet delay includes time spent waiting in the port output buffer for transmission on the link, the transmission time, and the propagation delay. It does not include port input buffer delays or node switching delays.

The routing table calculation is performed at the start of simulation, and then on each node or link failure or recovery event. In addition periodic updates are required as the delays experienced on each link may vary over time due to changing congestion levels.

The time period between updates is given in the “Routing Update Interval” field. Link delays are averaged over this period.

The Deviation Percent is based on the total link delay across all hops of a route.

### User-Defined Routing Tables

The User-Defined Routing protocol selects a route from 1 or more routes that have been manually entered by the user. Multiple alternate routes may be entered. Partial routes may also be entered which route the packet to some intermediate node. At the intermediate node one or more routes must be available to forward the packet towards the destination.

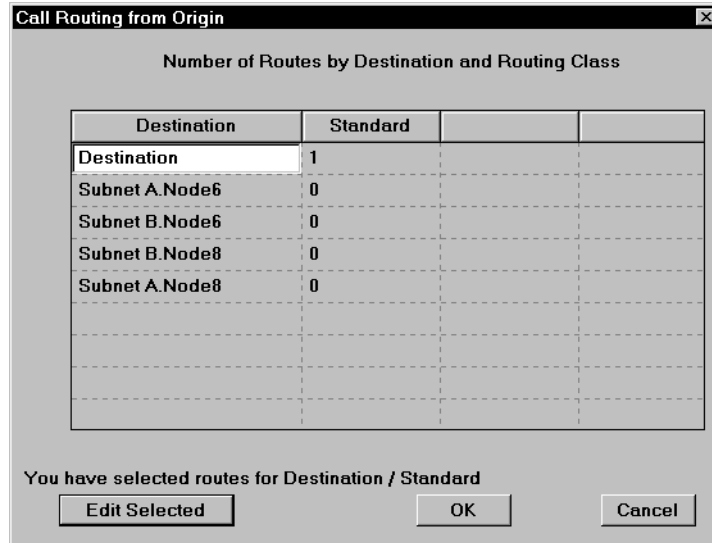
When a user enters a number of possible alternate routes, a selection criterion is required to determine which alternate to use. These criteria include:

- First Available
- Max Unused Bandwidth
- Min Delay
- Min Queue
- Min Sessions
- Random List
- Round Robin

A route in User-Defined routing is implemented by keeping a routing table at each node. Each table may contain a choice of single hops to use to reach some next node, or it may list several hops and so route the packet all the way to the destination. These may be regarded as implementations of Node-By-Node routing and Source-Node routing respectively. In addition, partial routes that only get part way to the destination are allowed.

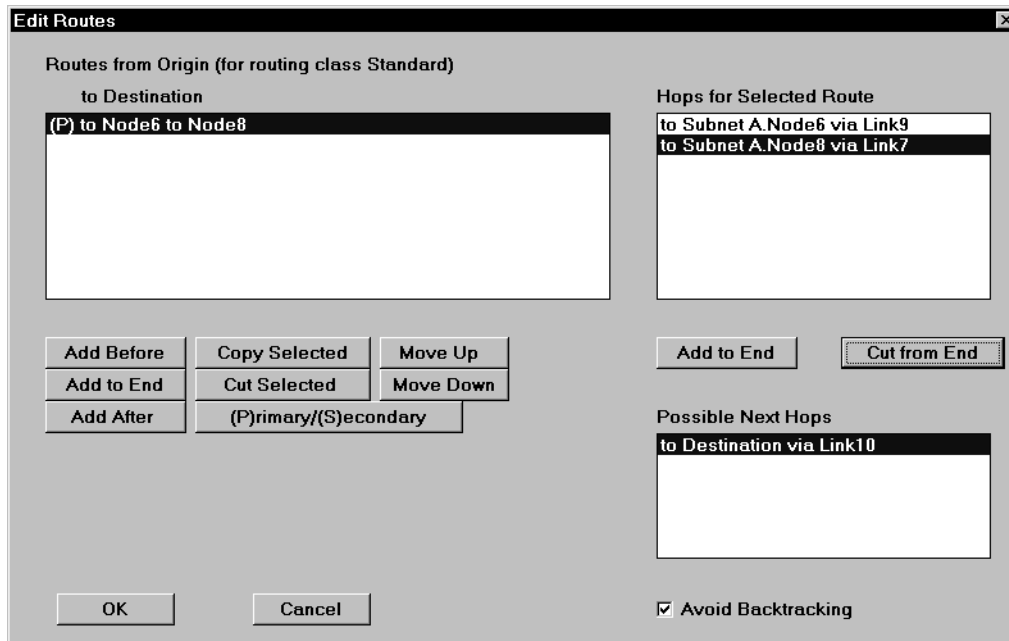
#### 4. Modeling Constructs

The routing tables are entered for each node by bringing up the node dialog box and then selecting the Packet Routing Table button. A Destination Selection dialog box then presents a table of destinations and routing classes. After highlighting a particular destination/routing class intersection, routes to that destination may be entered by selecting the “Edit Selected” button.



#### Destination Selection Dialog Box

A Route Definition dialog box is then presented. This allows a number of routes to the particular destination/routing class combination to be entered. The routes may completely reach the destination, or partially reach the destination.



### Route Definition Dialog Box

With User-Defined Table routing, both primary and secondary routes may be defined. When a routing decision is made, the primary routes are inspected and a route found. If no route can be found from the primary routes, then the secondary routes are evaluated. If still no route can be found, then the packet is counted as blocked.

The “Avoid Backtracking” check box has no impact on model execution. When a route is being defined, and with the box checked, links which return to a node already in the route will not appear in the “Possible Next Hops” list. With the box unchecked, all outgoing links from a node appear in the “Possible Next Hops” list.

Backtracking is useful for routing packets to special routers (especially on LANs) or validation servers. An example is for a WAN cloud service where all traffic must be routed at a single router outside the WAN cloud.

If there are several routes defined at a routing node, then when a packet requires routing some tie breaking mechanism is required. These include:

#### First Available

The routes are inspected in the order defined by the user and must pass the following tests:

- the hop limit for the routing class must not be exceeded.
- both the nodes and links on all hops must be up.
- the route must not return to a previously visited node.
- if a session setup packet is being routed, all links must be under their session limits.

## 4. Modeling Constructs

Note that buffer space and availability is not part of the tests. The first route that passes the tests will be selected and the packet directed to the selected buffer, even if this results in immediate blocking because of buffer overflow.

If a multihop route is picked for a packet at a routing node, then the packet carries with it the list of hops it must use in order to follow the route. At each intermediate node the packet only has to look at the established route and be placed in the appropriate port output buffer (if possible)—the routing protocol at each intermediate node does not have to be queried.

### 4.76.1 Max Idle Bandwidth

With maximum idle bandwidth selection, the alternate routes defined by the user are inspected and a sublist made of routes which are inside the hop limit, are available, do not visit previous nodes, and have all links inside their session limits (if a session setup packet is being routed). From this sublist the route with the most idle bandwidth is selected.

A route comprises one or more hops. The link on each hop has an average bandwidth use, averaged over the “Link Utilization Update Interval” period. The link therefore has an amount of unused bandwidth (equal to the total bandwidth less the used bandwidth). On each route one of the links will have the least amount of unused bandwidth on that route. If there are several routes, then there will be one link on each route that has the least available unused bandwidth. These “least bandwidth” links are compared and the route selected which owns the link which has the most unused or idle bandwidth. In other words, find the “worst” link on each route and take the route that has the best, “worst” link.

Note that the bandwidth figure that is used is the one which resulted from the last completed measurement based on the “Link Utilization Update Period” value. For instance, if the “Link Utilization Update Period” is set to 5 seconds then the utilization percentages for each link will be computed every 5 seconds—i.e., at 5, 10, 15, 25, etc., seconds into the simulation. If a max unused bandwidth routing decision is made at 22.396 seconds into the simulation, the utilization figures available at that time are those that were last made at 20 seconds.

If two or more routes are tied in terms of their maximum unused bandwidth then they will be selected for routing on a round-robin basis.

### 4.76.2 Min Delay

With minimum delay selection the alternate routes defined by the user are inspected and a sublist made of routes which are inside the hop limit, are available, do not visit previous nodes, and have all links inside their session limits (if a session setup packet is being routed). From this sublist the route with the least delay is selected.

A route comprises one or more hops. The link on each hop has an average delay, averaged over the “Delay Update Period.” The current value of average delay for all links on the route are totalled to give the total delay for the route. If there are several routes, then there will be one route which has the smallest total delay.

Note that the delay figure that is used is the one which resulted from the last completed measurement based on the “Delay Update Period” value. For instance, if



the “Delay Update Period” is set to 5 seconds then the average delays for each link will be computed every 5 seconds—i.e., at 5, 10, 15, 25, etc., seconds into the simulation. If a minimum delay routing decision is made at 22.396 seconds into the simulation, the delay figures available at that time are those that were last made at 20 seconds.

If two or more routes are tied in terms of their delay the hop in the table for the first one will be selected and then that hop will be placed after the remaining ones of the same delay in the table. This will result in round robin selection among tied table entries.

### 4.76.3 Min Queue

With minimum queue selection, the alternate routes defined by the user are inspected and a sublist made of routes which are inside the hop limit, are available, do not visit previous nodes, and have all links inside their session limits (if a session setup packet is being routed). From this sublist the route with the minimum queue is selected.

A route comprises one or more hops. The link on each hop has a port output buffer which queues the packets waiting to complete transmission. A multiaccess link has several port output buffers queueing packets, but only one of them would be used on a particular route. Across the route, one of the port output buffer queues will be longest. For each route there is therefore a longest queue. The longest queues between routes are compared and the route selected which has the shortest value; in other words, compare the “worst” queues between routes and select the route with the best “worst” queue.

The queue lengths used are instantaneous measurements made at the time of the routing decision.

If two or more routes are tied in terms of their minimum queue length, the hop in the table for the first one will be selected and then that route will be placed after the remaining ones of the same queue length in the table. This will result in round-robin selection among tied table entries.

### 4.76.4 Min Sessions

With minimum session selection the alternate routes defined by the user are inspected and a sublist made of routes which are inside the hop limit, are available, do not visit previous nodes, and have all links inside their session limits (if a session setup packet is being routed). From this sublist, the selected route is the one with the minimum number of sessions currently established.

If two or more routes are tied in terms of the number of sessions, the route in the table for the first one will be selected and then that route will be placed after the remaining ones of the same number of sessions in the table. This will result in round-robin selection among tied table entries. For instance, if “minimum sessions” selection were used but only messages (rather than sessions) were created, or if connectionless routing for sessions is set, then no sessions are ever setup across a route and so all routes would have 0 sessions. This would result in round-robin operation as described.

## 4. Modeling Constructs

### 4.76.5 Random List

With random list selection, the alternate routes defined by the user are inspected and a sublist made of routes which are inside the hop limit, are available, do not visit previous nodes, and have all links inside their session limits (if a session setup packet is being routed). A random pick is made from this sublist.

### 4.76.6 Round Robin

With round robin selection, the alternate routes defined by the user are inspected and a sublist made of routes which are inside the hop limit, are available, do not visit previous nodes, and have all links inside their session limits (if a session setup packet is being routed). The route at the top of this list is used, after which it is placed at the bottom of the list. This results in round robin operation.

The destination of the route may be an intermediate node partially completing the route to the destination, or an access point at a subnet/backbone boundary. At this point another routing decision may be required to carry the packet further. If this decision succeeds then the packet is carried. If this decision fails to find an onwards route then the packet will block at this point and optionally be retried from the origin.

**Reporting:** N/A

**Fields:**

#### Link Utilization Update Interval

The “Link Utilization Update Interval” is used in the routing protocols under the packet-switched model. Where a link utilization figure is required, it is averaged over this time period.

Each link has a utilization measurement. For example, if the update interval is 5 seconds, then the utilization figures are updated every 5 seconds. If a utilization figure is needed between update points, the figure from the last update point is taken. For instance, if a utilization figure was needed at 24.5 seconds with an update period of 5 seconds, the utilization calculated over the range 15 to 20 seconds would be used.

**Routing Update Interval** The time period between periodic routing table updates.

**Delay Update Interval** On user defined table routing with minimum delay selection, the time period over which the delay statistics are collected.

**Deviation Percentage** The percentage relative to the shortest route within which other routes are considered equivalent.

**Congestion Threshold Type** On minimum penalty routing, the penalty factors can vary by link utilization percentage or by link delay (seconds).

#### Connection-Oriented Routing For Sessions

#### 4.76 Routing Protocol: Packet

If checked, then session setup packets are routed in a connection-oriented manner. A virtual circuit is established across the network which is then followed by the data packets and any response or ACK packets.

If connection-oriented routing is off then all packets are routed on a datagram basis.

## 4. Modeling Constructs

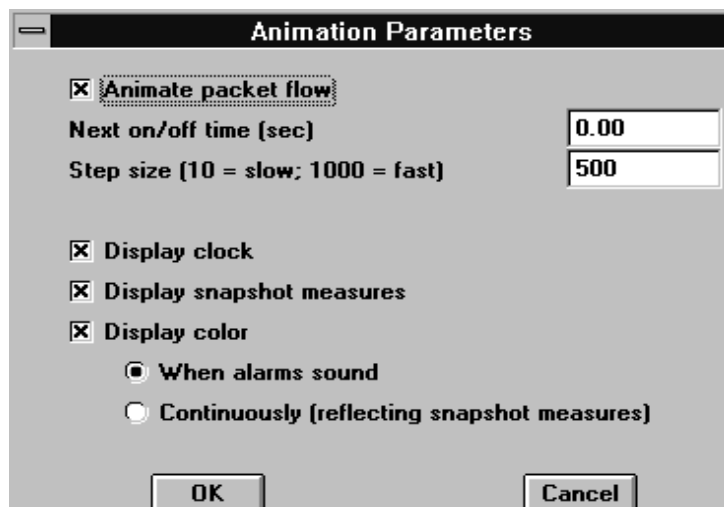
### 4.77 Simulation: Animation

**Purpose:**

To view the dynamic operation of the simulation model. Packet flow is animated along links. Call traffic is reflected by changing utilization figures next to nodes and links, and by indication of selected routes if tracing is turned on.

While the simulation is running different parts of the model can be viewed by panning around using the scroll bar controls, or by double clicking on subnet icons to enter the respective subnet and see what is happening inside it. Double clicking on the background in the subnet will return to the backbone level.

Because animation merely show frames entering or packets leaving a link, no simulation time passes while animation is shown. However, animation does take time on the computer running the simulation, and so a simulation typically runs several orders of magnitude faster with animation turned off.



Animation Parameters Dialog Box

**Creating:**

N/A

**Connectivity:**

N/A

**Editing:**

These parameters are set in the Animation Parameters dialog box and apply globally to all subnetworks and the backbone.

**Execution:**

When animation is turned off, COMNET III performs standard discrete event simulation. This means that upon completion of an event, it skips directly to the time of the next scheduled event and performs that event.

When animation is turned on, selected events are represented graphically. Although real time elapses during the graphical display of an event, no simulation time elapses.

For the packet-switched model, packet flow across nodes and links is animated. When a packet is transmitted across a link it is broken into transmission frames. Each frame is shown by a small red rectangle which moves from the transmitting

node icon across the arc to the link icon. When all frames have been transmitted then, after the propagation delay for the last one, the packet can be rebuilt from the frames and placed in the port input buffer of the receiving node. This is reflected by a small blue rectangle moving from the link icon across the arc to the receiving node icon.

The movement of these rectangles reflects the event of starting frame transmission, or of receiving the packet into a port buffer. These events are instantaneous in simulation time. The time duration of the movement of the rectangle should not be interpreted as reflecting a transmission delay—they really indicate that an instantaneous state transition has taken place. The time between a red frame rectangle starting to move and the time a blue packet rectangle starts to move is a better indication of link transmission time.

For circuit-switched call traffic, the default animation mode is to indicate the instantaneous link and node utilization as a number next to the respective link or node icon. As a call is routed, it takes bandwidth from nodes and links along the selected route and so their utilization is increased. When a call clears the bandwidth is given back and so the utilization is decreased.

The trace option shows additional animation for circuit-switched calls. Successful routing decisions will then be shown by highlighting the chosen route in green. Blocked routing decisions are shown in red.

**Reporting:**

N/A

**Fields:****On/Off**

A check box to indicate whether animation is on or off. This check box can be changed while the simulation is running.

If the model is saved with animation on then when the model executes it will be animated. Similarly it will not be animated if saved with animation off.

At the end of simulation the check box is changed back to its initial setting rather than remaining in its last changed value.

**Time Of Next Change**

The simulation time at which the current setting of the on/off check box will be toggled; i.e., the time at which, if it is off, it will turn on and vice versa.

This can be used skip “boring” parts of the animation and then turn on the animation just prior to some interesting event (such as a failure event).

**Step Size**

The speed of animation is controlled by the Step size parameter which can be set in the range 10 (which is slow) and 1,000 (which is fast).

Turning animation completely off with the On/Off check box is the fastest execution mode.

**Display Clock**

Checking this box will turn on the simulation clock in the lower right hand corner of COMNET III canvas.

**Display Snapshot Measures**

By checking this box, reporting snapshot measurements will be displayed if they

#### 4. Modeling Constructs

have been enabled under the Reports Snapshots.

##### **Display Color**

Checking this box will allow objects to change color based upon existing alarm condition thresholds.

**When Alarm Sounds:** A static change in object color when an alarm would occur.

**Continuously (reflecting snapshot measures):** This will allow for a continuous color change to be used to indicate the levels of snapshot measures themselves, based upon alarm conditions.

### 4.78 Simulation: Animation - Dynamic Color Change

In release 1.2, if you checked the “show alarms” checkbox, dynamic color changes would be used to indicate alarm conditions. Specifically, a color change indicated an alarm had sounded. It was a discrete indication, on or off.

In release 1.3, dynamic color changes can be more continuous. You can specify that color be used to indicate the levels of snapshot measures themselves. By default, a color range from blue to red is used to indicate a measure from 0% to 100%. By default, continuous color changes are shown only for snapshot measures that reflect a percent value, such as channel utilization.

Additionally, if you set an alarm threshold on a snapshot, the continuous color changes will then reflect the range from 0 to the threshold, again from blue to red. This effect applies to any snapshot that has an alarm set, whether percent valued or otherwise.

Note that, just like measures, continuous color changes reflect only the snapshots selected for display in the Select Snapshots dialog. Only one snapshot per class of objects can be selected for display. In other words, to show continuous color for channel utilization, you must

- 1) Go to Select Snapshots and turn on Channel Utilization for one or more links,
- 2) Select Channel Utilization as the display snapshot for links,
- 3) Go to Animation and be sure “Display color” is checked,
- 4) Select the “Continuously” radio button.

The controls of the Animation dialog can now be used to obtain any of the following combinations of snapshot display:

- Display measures (the little numbers over the icons) and change colors whenever an alarm sounds
- Display measures and reflect the measures in color changes
- Don’t display measures, but change colors whenever an alarm sounds
- Don’t display measures, but indicate the measures by color changes.

## 4. Modeling Constructs

### 4.79 Simulation: Parameters

**Purpose:**

Simulation control parameters affect duration of the simulation and the number of reports produced. Ultimately, the statistical accuracy and your interpretation of simulation results will be influenced by these parameters.

Simulation controls specify parameters such as the model warmup time, the number of times the model will be executed (number of replications) each time it is run and the length of time each replication will run.

Many factors influence the user's choice of warmup time, number of replications, and replication time. An excellent discussion of these topics is presented in *Simulation Modeling and Analysis* / Averill M. Law, W. David Kelton. 2nd ed. New York: McGraw-Hill, 1991.

**Creating:**

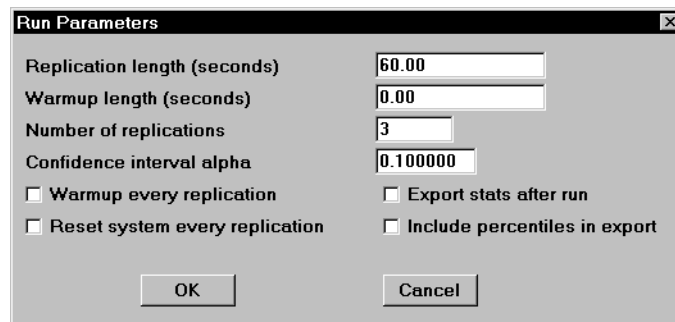
N/A

**Connectivity:**

N/A

**Editing:**

These parameters are set in the Simulate / Run Parameters dialog box.



**Run Parameters Dialog Box**

**Execution:**

See Fields below.

**Reporting:**

N/A

**Fields:**

**Replication Length**

Replication length is the number of seconds of simulated time during which statistics will be collected for each report. The aim with this parameter is to decide how long the model needs to operate until a sufficient number of events of interest have occurred such that the information which has been collected is statistically meaningful.

**Warmup Length**

Warmup length is the amount of simulation time which is allowed to pass until a replication starts and statistics are gathered. The aim is to let the model build up from having all resources idle and all queues empty to when the system reaches a steady state before starting to gather statistics on its operation. If statistics were



gathered right from the start, they could be distorted by low utilization rates and low congestion levels while the system came up to steady state.

#### Number Of Replications

The number of times the simulation will be repeated with different random number values selected so that a full statistical spread of different events in the model can be tested. At the end of each replication, a set of reports is written to file.

If the “Reset System After Every Replication” flag is set to yes then the network is emptied of traffic and all devices reset to their initial conditions; however, the random number seeds are not reset. Therefore the next replication executes with different starting seeds to the previous one and so is a statistically separate experiment.

A common simulation technique is to run a simulation a long period of time collecting statistical measures periodically as the simulation runs. COMNET III provides a set of reports at the end of each replication. Therefore, in order to achieve the equivalent of a long run, set the number of replications to the number of reports you want and do not reset the system for each replication. Not resetting the system means that the traffic that is established at the end of one replication will act as the starting condition for the next replication.

Many COMNET III models are designed to simulate operations under transient or dynamic conditions such as link outages or sudden peaks in traffic. In these cases the user is well served by repeating the transient condition a number of times and accumulating statistics over a number of these replications. In this case the system should be reset between replications so that traffic from one replication is cleared before starting the next replication. Then the network starts from an empty condition on each replication.

#### Warmup Every Replication

Indicates whether a warmup period should be included for every replication. If set to “No,” then only the first replication has a warmup period.

#### Reset System Every Replication

Indicates whether the model should be completely reinitialized from one replication to the next, or whether the ending conditions of one replication should form the starting conditions of the next. In either case, the random number seeds are not reset so that the next replication has different random number sequences.

#### Export Stats After Run

This option automatically produces an export file of all monitor statistics upon completing the simulation. The labeling of rows and columns in the export file have been improved to be much more readable. The export file can be imported directly into spreadsheet packages like Microsoft Excel for further analysis.

#### Include percentiles in export

This option when turned on, exports percentiles for every *Statistics...* button entry where the user selects the *Save Observations* checkbox. Percentiles are available everywhere observations are saved.

## 4. Modeling Constructs

### 4.80 Simulation: Tracing

**Purpose:** To trace model execution step by step for debugging and analysis purposes. If tracing is on, then, at every step, a detailed text description of what the model is doing is produced.

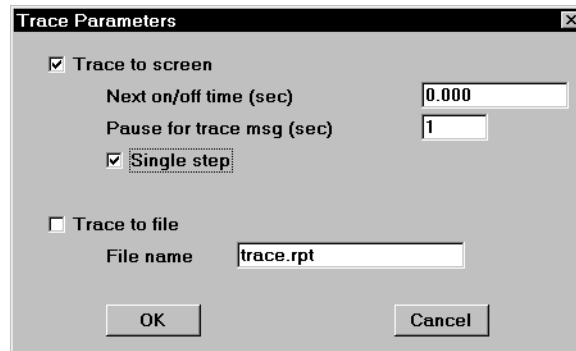
This is very useful if the model is behaving in an unexpected manner. The detailed sequencing of events and flow of control can then be watched.

**Creating:** Tracing is controlled from the **Simulate/Trace** menu option. This can be accessed both while editing the model description, and while executing the simulation.

If tracing to file is selected then a file with a user specified name will be created in a directory which has the same name as the model (i.e., the same directory as the reports). If a model is rerun with the same trace output file name, the older version of the file will be overwritten.

**Connectivity:** N/A

**Editing:** The trace file is in ASCII format and so can be loaded with any text editor. Note that trace files can become very large.



**Trace Parameters Dialog Box**

**Execution:** N/A

**Reporting:** N/A

#### **Fields:**

#### **Destination**

The trace messages can be sent to screen or file (or both). If sent to screen, then they appear on the scrolling status bar at the bottom of the screen.

If sent to file, then the name of the file is required. It is placed in the model results directory. If a file of the same name exists, it will be overwritten.

#### **Next On/Off Time**

The simulation time at which to toggle the tracing. If tracing is presently off, then

turn it on at the specified time, and vice versa.

**Pause For Message**

When the model is running and if trace messages are sent to screen, then the messages may pass so fast that there is no chance to read them. This field specifies how many seconds each trace message should remain on the screen. One overall impact is the model is then throttled to run at this speed.

**Single Step**

If the “Single Step” box is checked then the model moves from one trace message to the next and then waits for user input (a mouse click on a Continue dialog box) before continuing.

When the Continue dialog box is waiting for input, the **Simulate/Trace** menu option is still accessible so that single stepping can be turned off and other trace controls accessed.

## 4. Modeling Constructs

### 4.81 Sockets

*Purpose:*

The remote socket performs a similar function for clustering sources on a node for the common situation where there are many sources attached to the same node. However, the socket has more significance because it allows a group of sources to share parameters for packetization and assembly delays. A socket can be specifically identified as a message destination and when it is, its assembly time parameter applies at the attached node. The sockets also contain parameters for the packet size and window size that are used in a protocol's negotiation of these sizes, depending on the source and destination constraints.

Sockets are optional for any node or source. With no socket present, the sources will use the packet and window size specified by the protocol and there will be no mechanism to model an assembly time nor no option to model packetization that doesn't utilize the processor.

Socket Parameters

Parameter set name:

For messages destined to this socket

Assembly delay:  ..

Assembly uses processor

For sources on this socket

Packetization:  ..

Override other packetization parameters

Packetization uses processor

Processor swaps after each packet

Transport/Network layer size limitations for

	Messages to this socket	Sources on this socket
Window (Pkts)	<input type="text" value="1"/>	<input type="text" value="1"/>
Packet (Bytes)	<input type="text" value="65535"/>	<input type="text" value="65535"/>

OK Cancel

The sockets have an extendable list of parameter sets that contain the following parameters.

### **Packetization**

All traffic sources and traffic commands have a parameter for setting the packetization time for the packets generated by that source or command. The packetization time is the time required to build a packet from the message and set appropriate fields in its header. This is typically used to model transport layer protocol processing.

Because it is common that all sources using the same protocol stack will experience similar packetization delay, the socket's packetization delay provides a useful alternative that allows the packetization delay to be specified in one place and apply to all sources attached to that socket. The packetization delay specified at

the socket includes options for modeling the delay without utilizing the node's processor or specifying that delay in terms of processor cycles or milliseconds.

Because there is also a packetization time specified on the source or command, COMNET will use the sum of both packetization times when sockets are present. It is recommended, however, to specify the packetization delay at the socket when the socket is used.

The socket provides an additional option for how the processor handles the packetization with a large window when there are other applications that are pending at the same time. One option is for the processor to swap the current application and begin working on a different application as soon as one packet is created, while the other option is for the processor to continue working on the same message until the flow-control window or the rate control prevents any further packets from being created. Typically the other applications waiting to be processed will be other instances of messages so that how quickly the processor relinquishes the processing of a particular message instance can have a significant impact on that message's overall delay.

## ***Assembly***

Packet assembly time complements the packetization time but applies to the destination node. If a message source has a destination of a specific socket that is on a node, then that socket's assembly time is used to calculate the delay to process that packet to rebuild the message at the destination. The assembly time has an option to model this delay without utilizing the destination node's processor although it is most useful when such utilization does occur. Also, the socket provides an option to interpret the assembly time in terms of milliseconds or cycles.

There is no alternative way to specify an assembly time without using sockets.

## ***Socket-Based Protocol Constraints***

As discussed in the protocol section, the actual packet size and window size used for a protocol can be modified based on the constraints at the source and destination. Recall that the protocol definition itself is globally available and may exist on many sources throughout a model. The socket can refine the packet size and window size in two ways. First, there are parameters to specify the constraints for packet and window sizes for traffic leaving the node through any sources attached to this socket. Secondly, there are parameters for these constraints for traffic arriving to this node from other sources with destinations of this specific socket.

When sockets are involved at the source and destination ends, the packet size chosen will be the minimum packet size from the protocol's field and the source packet size on the socket at the source end and the destination packet size at the socket at the destination end. (Note, however, that the sockets are optional at either end.) The window size for this connection will be the minimum window calculated in bytes from each of these three places.

## 4. Modeling Constructs

### 4.82 Statistics: Export File

Most of the summary statistics gathered during the simulation for use in generating reports can be exported to a tab-delimited, ASCII file for post-processing, typically with a spreadsheet program like Microsoft Excel. Each performance measure in the file includes the minimum, maximum, sum, sum-of-squares, and number of observations for each replication of the simulation.

	A	B	C	D	E	F
1	Acme	Thu May 16 15:18:04 1996				
2						
3						
4						
5	Title	Units	Mean	Standard Deviation	Sum	Sum of Squares
6	Node Input Buffer Level for Boston1	level	0	0	0	0
7	Node Output Buffer Level for Boston1	level	0.000521333	0.244853143	0.03128	3.5972
8	Storage Utilization for Boston1	bytes	0	0	0	0
9	Node Processor Utilization for Boston1	busy processors	0.0005	0.022355089	0.03	0.03
10	Node Input Buffer Level for DC1	level	0	0	0	0
11	Node Output Buffer Level for DC1	level	0.0016018	0.362090517	0.096108	7.866292
12	Storage Utilization for DC1	bytes	0	0	0	0
13	Node Processor Utilization for DC1	busy processors	0.003	0.054690036	0.18	0.18
14	Node Input Buffer Level for RouterB	level	1.887416667	13.27643019	113.245	10789.55641
15	Node Output Buffer Level for RouterB	level	17.91590901	41.87650891	1074.95454	124477.308
16	Storage Utilization for RouterB	bytes	0	0	0	0
17	Node Processor Utilization for RouterB	busy processors	0	0	0	0
18	Node Input Buffer Level for RouterDC	level	1.424333333	11.45913182	85.46	8000.425646
19	Node Output Buffer Level for RouterDC	level	13.34756002	35.18055808	800.853601	84949.74152
20	Storage Utilization for RouterDC	bytes	0	0	0	0
21	Node Processor Utilization for RouterDC	busy processors	0	0	0	0
22	Node Input Buffer Level for NYC	level	64.36667353	107.8207264	3862.000412	946102.662
23	Node Output Buffer Level for NYC	level	0.075741133	2.71780338	4.544468	443.531516
24	Storage Utilization for NYC	bytes	0	0	0	0
25	Node Processor Utilization for NYC	busy processors	0.81371966	0.389332731	48.82317959	48.82317959
26	LanB Channel Utilization	busy X channels	0.000932067	0.030515536	0.055924	0.055924
27	LanB Frame X Delay	seconds	0.000221043	4.15E-05	0.055924	1.28E-05
28	LanDC Channel Utilization	busy X channels	0.000705533	0.026552506	0.042332	0.042332

## 4.83 Statistics: Link

<b><i>Purpose:</i></b>	<p>To collect utilization and loading statistics on a link and present the results graphically.</p> <p>A point-to-point link is implemented on a full duplex basis and so there is transmission capacity in both directions. Consequently there is utilization in both directions and so two graphs may be plotted.</p> <p>For contention based links (such as Aloha or CSMA/CD) a control channel may be specified. The contention channel and the control channel utilization may be separately plotted.</p>
<b><i>Creating:</i></b>	<p>In order to obtain graphs for link loading and utilization, collection and saving of statistics must both be turned on before the simulation is run. This is achieved by selecting the “Statistics” button on the link dialog box, picking the desired statistic from the list box, and editing the statistic detail to ensure that it is collected and saved.</p> <p>The simulation is then run.</p> <p>After the simulation is run the graphs can be viewed by picking the “Statistics” button on the link dialog box, selecting the statistic, and then picking the “View” button.</p>
<b><i>Connectivity:</i></b>	<p>Link statistics in terms of graphical output are available for the packet-switched bandwidth on all types of link.</p>
<b><i>Editing:</i></b>	<p>Click on the Statistics button on the link dialog box.</p>
<b><i>Execution:</i></b>	<p>When the simulation runs, if collection of link status has been turned on, link statistics are saved to the file “statfile” in the model directory. If “save observations” has been requested, the raw data are saved in the file <b>stattrc.out</b>. After the simulation has completed and when graphs are requested, it is the data in this file which is read and graphed.</p> <p>Running the simulation again will overwrite any previous statfile and <b>stattrc.out</b> in the model directory.</p>
<b><i>Reporting:</i></b>	<p>N/A</p>
<b><i>Fields:</i></b>	
<b>Collect Statistics</b>	<p>A check box to indicate that statistics should be collected on link utilization.</p>
<b>Save Observations</b>	<p>A check box to indicate that the raw data should be saved to file so that they can be plotted after the simulation finishes.</p>
<b>View</b>	<p>View the graphs on the collected statistics. Plot parameters to control the graph scaling, etc., may be entered. (see <i>Statistics: Plot Parameters</i>)</p>

## 4. Modeling Constructs

### 4.84 Statistics: Message

<b>Purpose:</b>	To collect message delay and message in transit statistics and present the results graphically.
<b>Creating:</b>	<p>In order to obtain graphs for message statistics, collection and saving of statistics must both be turned on before the simulation is run. This is achieved by selecting the “Statistics” button on the message source dialog box, picking the desired statistic from the list box, and editing the statistic detail to ensure that it is both collected and saved.</p> <p>Real time graphs are produced as the simulation is run. Note that production of real time graphs takes computation resources and so slows down the model execution.</p> <p>After the simulation is run the (non real time) graphs can be viewed by picking the “Statistics” button on the link dialog box, selecting the statistic, and then picking the “View” button.</p>
<b>Connectivity:</b>	<p>Message statistic plots are available for messages which arise from message sources, response sources, and session setup sources.</p> <p>Plots are not available for message commands defined directly on a node.</p>
<b>Editing:</b>	Click on the Statistics button on the message, response, or session source dialog boxes.
<b>Execution:</b>	<p>When the simulation runs, if collection of message information has been turned on, message statistics are saved to the file “statfile” in the model directory. If “save observations” has been requested, the raw data are saved in the file <b>stat-trc.out</b>. After the simulation has completed and when graphs are requested, it is the data in this file which is read and graphed.</p> <p>Running the simulation again will overwrite any previous statfile in the model directory.</p>
<b>Reporting:</b>	N/A
<b>Fields:</b>	
<b>Collect Statistics</b>	<p>A check box to indicate that statistics (mean, minimum, maximum, standard deviation) should be collected.</p> <p>Applies to:Message Delay Messages In Transit</p>
<b>Save Observations</b>	<p>A check box to indicate that the raw data should be saved to file so that they can be plotted later in the COMNET III session.</p> <p>Applies to:Message Delay Messages In Transit</p>
<b>Plot Parameters</b>	<p>(see Statistics: Plot Parameters)</p> <p>Applies to:Message Delay</p>



## Messages In Transit

**On/Off**

For real time graphs, indicates whether the graph should be drawn or not.

Applies to: Real Time Message Delay  
Real Time Messages in Transit

**Initial Y Axis Minimum**

For real time graphs, the minimum y axis value to be plotted when the graph is first drawn. As the simulation runs, the graphs are auto-scaling.

Applies to: Real Time Message Delay  
Real Time Messages in Transit

**Initial Y Axis Maximum**

For real time graphs, the maximum y axis value to be plotted when the graph is first drawn. As the simulation runs, the graphs are auto-scaling.

Applies to: Real Time Message Delay  
Real Time Messages in Transit

**Initial X Axis Minimum**

For real time graphs, the minimum x axis value to be plotted when the graph is first drawn. As the simulation runs, the graphs are auto-scaling.

Applies to: Real Time Message Delay  
Real Time Messages in Transit

**Initial X Axis Maximum**

For real time graphs, the maximum x axis value to be plotted when the graph is first drawn. As the simulation runs, the graphs are auto-scaling.

Applies to: Real Time Message Delay  
Real Time Messages in Transit

## 4. Modeling Constructs

### 4.85 Statistics: Monitors

COMNET III offers many statistics request monitor and allows many more variables to be monitored through the statistics request buttons on various dialogs.

#### *Option to Store Averaged Observations*

The statistics request monitors enable the collection of raw observation data to be plotted or to be analyzed for percentiles or exported to other statistics tools. The option allows the sampled observations to be averaged observations over an interval.

If the interval is zero, then the monitors will collect raw observations for plotting. For many values such as utilization or buffer levels, there can be a tremendous number of observations—making the simulation run slow to store all those observations to the hard drive and then create observation files too large to manage.

If the interval is larger than zero, then the monitors will collect one sample per interval and that sample will be averaged (either time-weighted for values such as utilization, or tallied for values such as delay) over the interval. This can greatly reduce the number of samples that have to be stored in a file. It also adds an advantage by providing more meaningful plots, histograms, or percentiles because of the smoothing it provides. It is especially useful for monitoring utilization and buffer levels but it can also be helpful for plotting averaged delay for large numbers of messages.

#### *Monitor Statistics and Confidence Intervals*

The table for viewing monitor statistics has been improved to provide better labeling and precision. When multiple replications are available, confidence intervals are also provided. The desired confidence interval alpha is specified on the Run Parameters dialog, immediately after the number of replications. If monitor observations have been saved, percentiles can also be included in the table of monitor statistics.

#### *Export to Excel*

An improved export capability is also available for all statistical monitors. There is a new Run Parameters option that automatically produces an export file of all monitor statistics upon completing the simulation. The labeling of rows and columns in the export file have been improved to be much more readable. The export file can be imported directly into spreadsheet packages like Microsoft Excel for further analysis.

#### *Node Monitors*

Release 1.2 provides a statistics request button on nodes to provide the following statistics on the node:

##### **Processor Utilization**

This monitor measures the number of busy processors at any time. For a single processor node, the result is the utilization of the processor. Frequently the processor will be busy with short duration tasks and thus it is generally desirable to

collect this utilization averaged over a convenient interval.

### **Storage**

This monitor measures how much file space is used in this node's disk storage space. This monitor is useful for models that have read and write commands that change the size of files.

**Call Bandwidth Level (Utilization)** This monitor measures the call bandwidth allocated through this node. It is useful when modeling bandwidth-switched calls with call sources, and it gives the total bandwidth of all the calls using this node.

## ***Buffer Monitors***

### **Buffer Level**

The input and output buffers have a monitor for monitoring the buffer level. This monitor may be available for off-line analysis or real-time plotting. Since packets arrive and leave buffers very frequently, it is generally desirable to collect averaged buffer levels for plotting.

### **New Link Monitors**

Two new monitors have been added to monitor the link:

**Call Bandwidth Level (Utilization)** This monitor measures the call bandwidth allocated through this link. Call bandwidth is allocated on the link when a bandwidth switched call (from a call source) uses this link.

### **Frame Size**

This monitor measures the size of frames using this link. Since there are likely to be a large number of frames using a link during a simulation, this monitor should be averaged over an interval for best results.

### **New Message Destination Monitors**

There is a set of detailed protocol monitors available for each destination of a message command or a traffic source. These monitors are only available for messages with explicit destination lists, and they are not yet available for global commands.

### **Message Delivery and Message Transmission Delays**

The message delay is now measured from two perspectives: the delivery delay is the time required for the receiver to reconstruct the message while the message delay is the time required before the source closes the message command. Due to acknowledgements, retransmissions, or a close-sequence, the sender will generally close the message transmission later than the receiver will receive the message.

### **Window Size**

The window size monitor is only for the TCP/IP congestion window that changes over time based on congestion on the network.

### **Inter-Packet Interval**

This monitor measures the time between when new packets are created within a message. It is only useful for messages that generate multiple packets. It can be useful as a measure of the effectiveness of the protocol's flow control window or rate control mechanism.

### **IPackets in Retransmission Event**

This monitor is primarily useful for the sliding window and TCP/IP window option for "fast recovery" to avoid retransmitting a whole window when a packet

#### 4. Modeling Constructs

is blocked.

##### **Retransmissions for Blocked Packets**

This monitor measures the number of retransmissions required for the same packet before the packet is received by the destination without errors.

##### **Retransmission Timeout**

This monitor is useful for the sliding window and TCP/IP window option to calculate the retransmission time based on the round-trip time of the packets. However, it is only updated when a retransmission actually occurs.

##### **Round-Trip Time (Ack Delay)**

This monitor measures the ack delay which is set up to measure the round-trip time of a packet. It is most useful for sliding windows that generate acknowledgements for each packet. It also provides a simple measure of network congestion.

##### **Packet Size**

This monitor measures the size of the packets generated by the protocol. This monitor is probably most useful for messages that consist of one packet.

##### **Burst Size**

This monitor measures the main burst size from the traffic policy of the protocol when the traffic policy is used.

##### **Inter-Assembly Interval**

This monitor measures the interval between successive packets arriving at the destination. This is particularly useful for time-critical traffic such as voice or video transmissions where a near constant inter-assembly interval is desired. When modeling ATM networks, this measures the cell-delay variation that is one measure of quality of service.

## 4.86 Statistics: Plot Parameters

<i>Purpose:</i>	To display statistics collected on various measures (such as link utilization) after the simulation has run.
<i>Creating:</i>	The graphs of collected statistics are accessed via the statistics buttons on the links, traffic generators etc. Once the raw data have been saved then you may view the graphs by clicking on the view button.
<i>Connectivity:</i>	N/A
<i>Editing:</i>	<p>The data that is graphed is collected during the simulation run. to change the data the simulation needs to be rerun.</p> <p>Various interactive parameters are available to control the appearance of the graphs. These are listed below.</p>
<i>Execution:</i>	N/A
<i>Reporting:</i>	N/A
<b>Fields:</b>	
<b>Observed Data</b>	Plot the data points that have been observed.
<b>Smoothed Data</b>	Plot smoothed data where each data point represents an average of the observations over an interval. In the case of delay measures, the number of points to plot determines the averaging interval (more points means a smaller interval). If there is no delay observation on an interval, zero is plotted. In the case of level measures, the user specifies the averaging interval.
<b>Number of Points</b>	The number of points to plot.
<b>Start Time</b>	The simulation time at which to start graphing.
<b>Stop Time</b>	The simulation time at which to stop graphing.
<b>Averaging Interval</b>	Applies only to level measures (e.g., number of busy channels, number of messages in-transit, buffer level). Specifies the time interval over which to compute the time-weighted average level that is to be plotted. The point plotted at time $t$ is the average level over the interval $t - \delta$ , where $\delta$ is the averaging interval. Note that the minimum value for $\delta$ is the interval covered by the plot (Stop Time - Start Time) divided by the number of points.
<b>Histogram</b>	Plot a histogram on the data that has been collected.
<b>Number Of Bins</b>	The number of bins to plot.
<b>Minimum</b>	The lower data limit for the first bin.
<b>Maximum</b>	The upper data limit for the last bin.

## 4. Modeling Constructs

### 4.87 Subnetwork

#### *Purpose:*

The subnetwork serves two purposes.

Firstly, it aids in managing the display of a large network. Rather than having one flat picture that has either very small icons or requires extensive scrolling, various sites or domains in the network can be placed in a subnetwork which is represented by just one icon at the backbone level.

Secondly, different subnetworks can be used to represent different domains in a network which operate under different routing protocols. For instance, a user site may be running a VAX network running DECNET which would run a link state shortest path algorithm. This may have one or more gateways into an X.25 backbone which operates under minimum hop. By making the DECNET a subnetwork in COMNET III, it may then have a routing protocol defined for it which is different from the backbone (and other subnetworks).

#### *Creating:*

To create a subnetwork use the subnetwork tool on the palette and drag a subnetwork onto the backbone layout.

Double clicking on the icon will then enter the subnetwork. Alternatively the subnetwork icon can be highlighted and the Edit/Enter menu option selected.

Upon entering a new subnetwork there are no network components in the new subnetwork. At least one access point must be created inside the subnetwork (by using the access point tool on the tool palette) before any connections can be made to the subnetwork. An access point can also be added at the network level by dragging the access point from the palette and dropping it on the edge of a subnetwork icon.

To leave the subnetwork and return to the backbone, double click on the background of the subnetwork. Or select the Edit/Leave menu option.

Subnetworks cannot be created inside other subnetworks.

#### *Connectivity:*

Within the backbone network, links are used to connect a backbone node to a subnetwork access point.

Within the subnetwork, an access point is connected to a node on a one to one relationship.

In fact, the access point and the node to which it is connected are synonymous. Editing the access point is the same as editing the node to which it is connected. Therefore only one node can connect to an access point. The main reason for the existence of access points is that their icons can be seen both in the subnetwork and in the backbone.

#### *Editing:*

To edit the parameters for a subnetwork, highlight the subnetwork icon and then select the **Edit/Detail** menu option. The dialog box for subnetwork parameters is then presented.

Note that double clicking on a subnetwork icon enters the subnetwork rather than presenting its dialog box.

Subnetwork Dialog Box

**Execution:**

When the model executes, traffic may originate in a subnetwork (or the backbone) with a destination in either the same subnetwork or a different subnetwork.

If the destination is in the same subnetwork (or backbone) the routing protocol for the subnetwork will be used to reach the destination. If no direct route available in the subnetwork then the traffic will block, even if there is some route that leads out through an access point through the backbone and back into the subnetwork at some other point. In other words, once the routing decision is local only the local routing algorithm will be used.

If the destination is not in the same subnetwork, the routing protocol will route to the “closest” access point to the origin. Once the traffic is at the access point the routing decision will then be handed over to the backbone routing algorithm to determine how to get closer to the ultimate destination.

“Closest” is measured in determines of the routing metric in use. If the minimum hop metric is used then it will be the access point the fewest hops away. If the minimum penalty metric is used then it will be the access point that is reached with the least total penalty.

**Reporting:**

N/A

**Fields:****Name**

The name of the subnetwork.

**Icon**

The icon with which to display the subnetwork.

**Traffic Scale**

To globally scale all of the traffic in a subnet or the backbone, set the scale parameter on the backbone detail or subnet detail dialog boxes. To edit the backbone detail, click on **Define/Backbone Detail**. To edit the detail for a subnet, select the subnet and then click on **Edit/Detail**. If you are inside of a subnet, you can still edit detail for the subnet by clicking on **Edit/Parent**.

The default value for the scale parameter is 1. To double the traffic in a subnet or the backbone, set the scale to 2. To halve the traffic in a subnet or the backbone,

#### 4. Modeling Constructs

set the scale to 0.5. The scale parameter applies to the interarrival time distributions for all traffic sources in a subnet (or in the backbone). When a value is drawn from an interarrival time distribution, the value is scaled by dividing by the scale parameter for the subnet where the traffic is originating. Thus, a large scale factor increases the arrival rate by making the interarrival times closer together and conversely for a small scale factor. The scale factor has no effect on message sizes or holding times.

**Call Routing Protocol**      (*see Routing Protocol: Call*)

**Packet Routing Protocol**      (*see Routing Protocol: Packet*)

#### **Link Utilization Update Interval**

Used by those Packet Routing Protocols which require a link utilization measurement. This is the period over which the average is taken.

When required, Call Routing Protocols use instantaneous measurements of link utilization and so this parameter is not used in the call model.



## 4.88 Traffic Policing: ATM

The ATM version of the traffic policy uses two burst measurements of the leaky bucket

(Generalized Cell Rate Algorithm, GCRA) type. The algorithm has been generalized to allow the algorithm to measure kilobytes or kilobits instead of packets although the true GCRA in ATM measures only the packets (which model ATM cells).

The screenshot shows the 'ATM Traffic Parameters' dialog box. The 'Name' field is set to 'DEFAULT'. The 'GCRA 1: Conformance' section has a 'Rate (/sec)' of 100.000000, a 'Limit burst' of 500.000000, and 'CLP counted' set to 'CLP = 0+1'. The 'GCRA 2: Sets CLP' section has a 'Rate (/sec)' of 50.000000, a 'Limit burst' of 500.000000, and 'CLP counted' set to 'CLP = 0'. In the 'General Options' section, 'Conformance' is set to 'GCRA 1 Only', 'Burst units' is set to 'Packets', and 'Set CLP' is set to 'Use Algorithm'. The 'Never Discard Packets' checkbox is unchecked. 'OK' and 'Cancel' buttons are at the bottom.

One of the bursts generally is used for measuring the conformance of the cell to the required rate. If the conformance burst exceeds its threshold, the cell will be immediately rejected. There is an option to use both bursts for conformance testing so that both bursts have to be less than their limits before the cell can be accepted.

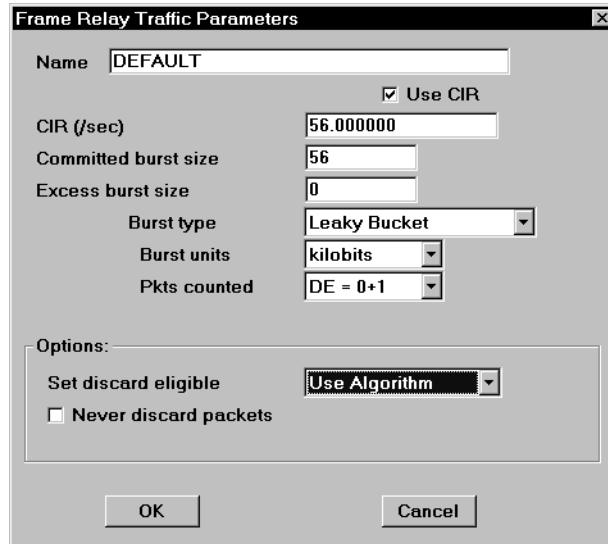
The second burst is used to determine whether the accepted cells will be flagged to be eligible for discarding at a buffer. In ATM the flagging is done and a bit marked CLP (cell loss priority) and the cell will be preferentially discarded at a buffer if its CLP is set (CLP=1).

As in frame-relay, there are options to force all cells to be CLP=1 or CLP=0, and to be sure that the algorithm never rejects any cells. These options may be useful for cases where these conditions are known to be true and it is not required to set up the algorithm to guarantee this.

## 4. Modeling Constructs

### 4.89 Traffic Policing: Frame Relay

The frame-relay version of the traffic policy uses a single burst measurement with two thresholds: one for setting discard-eligible (DE) packets and the other for immediately blocking the packet for exceeding the excess rate.



The screenshot shows a dialog box titled "Frame Relay Traffic Parameters". It contains the following fields and options:

- Name: DEFAULT
- Use CIR
- CIR (/sec): 56.000000
- Committed burst size: 56
- Excess burst size: 0
- Burst type: Leaky Bucket
- Burst units: kilobits
- Pkts counted: DE = 0+1
- Options:
  - Set discard eligible: Use Algorithm
  - Never discard packets

Buttons for "OK" and "Cancel" are located at the bottom of the dialog.

The burst measurement has options for measuring the burst in terms of packets, kilobytes (1024 bytes), or kilobits (1000 bits). The burst algorithm can be a sliding window, jumping window, or a leaky bucket algorithm. The burst is measured over an interval that can either be defined explicitly or inferred from a CIR parameter. The two thresholds are the committed burst size ( $B_c$ ) and the excess burst size ( $B_e$ ). Packets are flagged DE if the burst exceeds  $B_c$  and the packet is immediately rejected when the burst exceeds the sum.

There are options to have all packets flagged DE or never flagged DE, or to prevent any packets from being rejected immediately by the algorithm. These options are useful when the algorithm is not known but some constraint assures these conditions. One example is a type of traffic that is intentionally always flagged DE. Another example is the case where it is known that that source can not generate traffic exceeding the burst contract.

## 4.90 Traffic Source: Application

### *Purpose:*

An application source models the loading of some software task on an end system. Examples might be a transaction processing task on a host mainframe, a database access on a database server, or printing a spooled print job on a print server.

The application comprises a sequence of commands which must be executed. These may be any combination of read, write, process, transport, answer, or setup commands as required by the application.

It is each command which requests some resource to be allocated in order for the command to execute. The description of each command execution is found in the respective command section.

It is the task of the application to schedule when the commands are to be executed, and to control command sequencing. Consequently the application parameters are all related to scheduling and sequencing, and not to the detailed command description. The command description is described via the node dialog boxes, or via the global command dialog boxes.

Multiple concurrent applications can be scheduled on the same node. It is up to the node to determine how it will respond to simultaneous application requests.

### *Creating:*

To create an application pick the application tool from the palette and drag a new application onto the background. Or use the create menu option to create a new application. Once an application is drawn on the background it must be connected to a node using one of the arc tools. Once created and connected, the application icon can be double clicked to bring up the details of the application.

### *Connectivity:*

An application can only run on Computer & Communication (C&C) nodes, Computer Group nodes, and Router nodes. Connecting an application to the ATM node is not allowed.

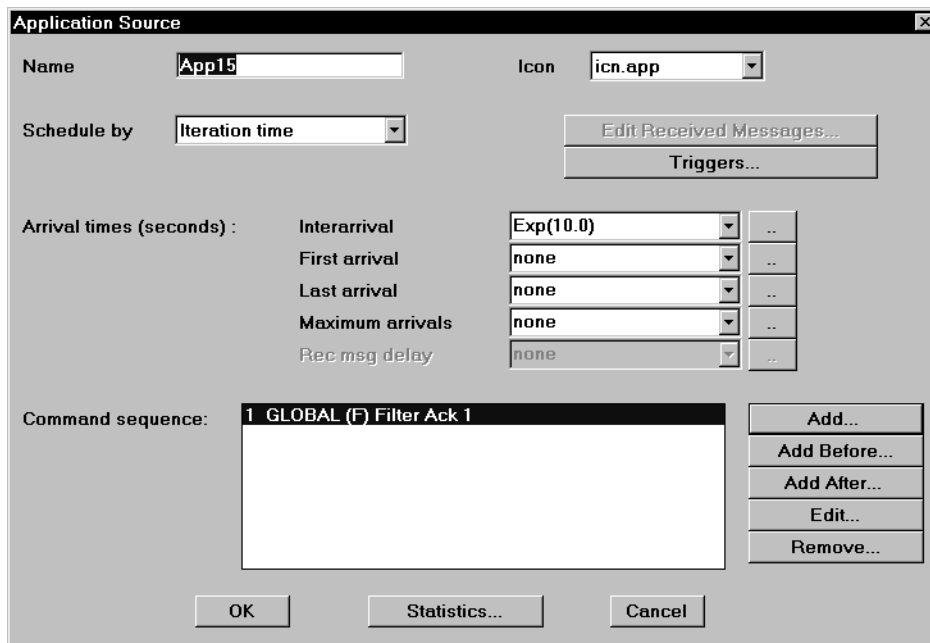
Connecting an application to a Computer Group node is the same as connecting an application to each of the nodes in the computer group. In other words, if there are 10 nodes in the computer group then connecting an application to the group icon is the same as connecting 10 applications to the group members.

Once an application is connected to a node, it can then “see” the commands that have been defined on that node. Such commands can then be referenced in the command sequence of the application. In addition, globally defined commands can also be referenced. Consequently, it is easier to first define the commands on a node and then the command sequence on an application source connected to the node.

### *Editing:*

Either double click on the application icon, or highlight the icon and select the Edit/Detail menu option. The dialog box for editing the application detail is then presented.

## 4. Modeling Constructs



Application Dialog Box

### *Execution:*

When an application executes, it requests the node on which it is running to execute the first command in the application command sequence, and then the next, and so on until all of the commands have completed. The application is then complete.

Details of how a specific type of command executes may be found in the respective command section.

Details of how the node arbitrates between concurrently scheduled applications may be found in the respective node section.

In terms of scheduling, the application depicted on the user interface may be thought of as an application “prototype.” The application prototype has scheduling requirements defined for it. When the scheduling requirements are satisfied, a new instance of the application can be created which will then interact with and load the node on which the instance runs. The application prototype does not load or consume resources from the node to which it is attached.

For time-based scheduling, the application prototype waits until the appropriate time point and then creates another application instance.

If an iteration period is specified then the time is from the start of one instance of the application to the start of the next instance. There is no constraint that multiple instances of the same application cannot overlap.

If a delay time is specified then the time is from the end of one instance to the start of the next. In this case there will only be ever one instance of a particular application in existence at a particular point in time as the first one must complete before the next one can start.

For message-based scheduling, when the application prototype has received the requisite messages, a new application instance can be created. Message requirements can include multiple messages and wildcards. For a fuller description of message based scheduling *see Received Message Scheduling*.

Once an application instance has been created, it is then considered for execution by the node on which it is running. Determining which instance to execute next and whether to single task or multitask is a decision which is left to the node.

*See Node: Computer & Communications (C&C Node)*

Node: Computer Group

**Reporting:**

Node: Application Delays

Other reports are affected by the traffic generated from within an application.

**Fields:**

**Name**

An alphanumeric field to identify the application.

**Icon**

The name of the icon used to represent the application.

**Schedule By**

Application instances can be scheduled by either "Delay Time," by "Iteration Time" or by "Received Message."

Delay time scheduling uses the given times as the interval between the completion of one instance of the application and the start of the next instance of the application.

Iteration time scheduling uses the given times as the interval between the start of one instance of the application and the start of the next instance of the application. Overlapping instances of the same application can exist with this scheduling.

If only one instance of an application is required in a replication, one way to achieve this is to use iteration time scheduling and to set the "Interarrival Time Distribution" to "None," the "Last Arrival" to "None," and the "First Arrival" to the time when the single instance is required.

Received message scheduling requires that one or more messages must arrive at the node where the application prototype exists for new instances of the application to be created. In fact the scanning for received messages is based on the incoming "Message Text." Multiple messages may be waited for, and message text may be scanned by wildcard.

*See Received Message Scheduling*

**Interarrival Time Distribution**

For time-based scheduling, the "Interarrival Time Distribution" is used to make a statistical pick for the time between application instance creation.

Any of the statistical distributions found in COMNET III may be used.

*See Distributions sections.*

**First Arrival**

For time-based scheduling, the time when the first application instance should be started. If this is left at "None," the first application instance will be created at a

#### 4. Modeling Constructs

time picked uniformly at random between 0 and a random pick from the “Interarrival time Distribution.”

##### **Last Arrival**

For time-based scheduling the time when the last application instance will be started. If this is left at “None,” application instances will continue to be created up to the replication length. Otherwise, the last application will be started at this time.

##### **Max Arrivals**

For time-based scheduling, the maximum number of application instances that can be created from the application prototype in the replication. If this is left at “None,” there is no limit on the number of instances.

##### **Received Message List**

If received message scheduling is used, the message texts which must be received in order to create a new application instance must be listed.  
*See Received Message Scheduling.*

##### **Received Message Delay**

For received message scheduling, the time after the received message list is satisfied when the command sequence starts.

##### **Command Sequence**

When an application instance has been created, it will then demand execution on the node to which it is attached. Demands for execution are, in fact, requests that a sequence of commands are executed by the node processor one after the other. It is up to the node processor to determine when it is available to forward any particular command.

On the application prototype, the sequence of commands that are required are listed in order in the “Command Sequence” list box. Against each command the number of times it is to be executed is specified.

When the last command in the sequence has completed then the application instance is counted as completed.

The commands can be any sequence of commands that have either been define locally on the host node, or globally in the global command list.

## 4.91 Traffic Source: Call

### *Purpose:*

A Call Source is used to model the generation of circuit-switched traffic.

Separate Call Sources could be used to model different types of traffic from the same origin—for instance one for voice traffic, one for data traffic and one for fax traffic.

The Call Source specifies the volume of a particular call type to be generated. The routing of each call instance is handled by the “Call routing Protocol.”

### *Creating:*

To create a Call Source, pick the Call source tool from the palette and drag a new call icon onto the background. Or use the create menu option to create a new call icon. Once a call icon is drawn on the background it must be connected to a node using one of the arc tools. Once created and connected, the call icon can be double clicked to bring up the details of the call.

### *Connectivity:*

A Call Source can only be connected to a Computer & Communications node.

### *Editing:*

Either double click on the Call Source icon, or highlight the icon and select the Edit/Detail menu option. The dialog box for editing the Call Source detail is then presented.

Note that the required bandwidth for this call is in the routing class parameter for the call source.

The screenshot shows the 'Call Source' dialog box with the following settings:

- Name: Call14
- Icon: icn.call
- Schedule by: Iteration time
- Priority: 1
- Routing class: Standard
- Arrival times (seconds):
  - Interarrival: Exp(10.0)
  - First arrival: none
  - Last arrival: none
- Duration (min): Nor(5.0,2.0)
- Dest type: Weighted list

Buttons: Triggers..., Edit Destination List..., OK, Cancel

**Call Source Dialog Box**

### *Execution:*

The Call Source may be thought of as a prototype call which will periodically create new call instances for the model to route. It is the task of the prototype to check when the scheduling requirements for a new call instance have been met and so create the new instance.

The only scheduling method provided is by “Iteration Time.” This is the time from the start of one call instance to the time of the next call instance. Conse-

## 4. Modeling Constructs

quently, over the period of a replication, new call instances will be created.

Once a call instance has been created it is up to the call routing protocol to route the call to its destination.

### **Reporting:**

- Calls: Blocked
- Calls: Disconnected
- Calls: Preempted
- Calls: By Node
- Calls: By Link
- Calls: Utilization by Node
- Calls: Utilization by Link

### **Fields:**

#### **Name**

An alphanumeric field to identify the Call Source.

#### **Icon**

The name of the icon used to represent the Call Source.

#### **Interarrival Time Distribution**

Used to make a statistical pick for the time between call instance creation.

Any of the statistical distributions found in COMNET III may be used.  
*See Distributions sections.*

It would be typical for the interarrival time to be based on an Exponential distribution, which results in call generation according to a Poisson process.

#### **First Arrival**

The time when the first call instance should be started. If this is left at “None,” the first call instance will be created at a time picked uniformly at random between 0 and the mean of the “Interarrival Time Distribution.”

If only one instance of a call is required in a replication, one way to achieve this is to set the “Interarrival Time Distribution” to “None,” the “Last Arrival” to “None,” and the “First Arrival” to the time when the single instance is required.

#### **Last Arrival**

The time when the last call instance will be started. If this is left at “None,” call instances will continue to be created up to the replication length. Otherwise, the last call will be scheduled at this time.

#### **Duration**

The time a call instance is to last.

Any of the statistical distributions found in COMNET III may be used.  
*See Distributions sections.*

A call instance is a demand to hold an amount of bandwidth between an origin and destination for a given duration.

It is common to quantify voice traffic in terms of Erlangs. The Erlang load is not directly entered into COMNET III. Rather the call interarrival time and the call duration are entered, which are related to the Erlang load by the formula:

$$\text{Erlang Load} = \text{Mean Holding Time} / \text{Mean Interarrival Time}$$



The holding time and interarrival time must be specified in the same time units. Note that the Erlang load and the Erlang distribution are not the same type of object and should not be confused.

Voice call duration is usually based on a Normal distribution.

### Priority

The priority field is used to set the priority of each call instance. In the circuit-switched model priority is used when preemption is specified. This enables higher priority calls that cannot find a route to preempt lower priority calls and so be routed. Depending on the routing class settings, the lower priority call may then be reattempted.

The preempting call does not have to share the same origin or destination as the preempted call. The only requirement is that the preempting call requires bandwidth on some node or link common to its desired route and the preempted call.

Higher priority values reflect higher priority (i.e., priority 1 is lowest priority).

### Call Routing Class

Pick from list (*see Routing Class: Call*)

### Destination Type

The destination of the call. A call destination must be another C&C node. Multicast destinations for calls are not allowed.

The destination type may be picked from:

**Random List** (see Destination: Random List)

**Random Neighbor** (see Destination: Random Neighbor)

**Multicast List** (see Destination: Multicast List)

**Weighted List** (see Destination: Weighted List)

### 4.92 Traffic Source: Message

***Purpose:***

A Message Source is a message generator which is used to send messages from origin to one or more destinations. Packet routing for Message Sources is always on a datagram basis.

Message sources are useful for modeling many forms of data transport, including general transaction oriented traffic, e-mail, and file transfers.

A Message Source may be created and represented directly on the user interface. Beneath the surface, the Message Source is translated into an application which executes a single transport command. In other words, creating a Message Source is the same as:

- Defining a transport command on a node
- Creating an application and attaching it to the node
- Calling the transport command from the application.

Directly creating and editing a Message Source is easier than this three stage process, because it is not necessary to define a separate transport command on the node.

***Creating:***

To create a Message Source, pick the Message source tool from the palette and drag a new message icon onto the background. Or use the create menu option to create a new message icon. Once a message icon is drawn on the background it must be connected to a node using one of the arc tools. Once created and connected, the message icon can be double clicked to bring up the details of the message.

***Connectivity:***

A Message Source can be connected to a Computer & Communications node, to a Computer Group node, or to a router node. If connected to a Computer Group, then internally to the model there is a Message Source for each instance node.

If the destination of a Message Source is a Computer Group node then any particular message will only be sent to one instance node in the Computer Group. A uniform random pick is used to determine which instance node to use. However, if a Computer Group node is included in a multicast destination list then the message will be sent to all instance nodes in the Computer Group.

***Editing:***

Either double click on the Message Source icon, or highlight the icon and select the Edit/Detail menu option. The dialog box for editing the Message Source detail is then presented.

**Message Source Dialog Box**

**Execution:**

A Message Source is executed as an application with one transport command.

For scheduling logic, see *Source: Application*.

Delay Time scheduling is not supported.

There is no “Max Arrivals” field.

For packet generation and transmission, see Command: Transport.

**Reporting:**

- Messages: Delays For Message & Response Sources
- Packets: Statistics For Message & Response Sources

**Fields:**

See “*Traffic Source: Application*” and “*Command: Transport Message*” for field descriptions.

## 4. Modeling Constructs

### 4.93 Traffic Source: Packet Flow/Packet Rate Matrix

*Purpose:*

COMNET III now includes a new mechanism, the **Packet Rate Matrix** for directly generating packet traffic without scheduling applications and without producing messages with transport commands. Such a mechanism is useful when data are available on packet rates and sizes, but not much is known about the internals of the application producing a stream of packets. The model has a list of such background packet flows. Each packet flow has an origin, a destination, a packet interarrival time and size distribution, application, and protocol. The application and protocol types are used solely for reporting purposes--they do not affect packet scheduling in any way.

AutoBaseliner can now obtain end-to-end packet statistics through new interfaces to Expert Sniffer and Frontier and Axon RMON data. Each end-to-end packet stream identified by AutoBaseliner is modeled in COMNET III by a background packet flow. For a typical imported packet stream, the available information includes the origin, destination, packet rate, and average packet size. In some cases, there is a breakdown by application and protocol. COMNET III automatically builds a background packet flow for each imported packet stream. You can edit the list of packet flows to make any adjustments to the flow models built by COMNET III.

*Creating:*

The Packet Rate Matrix is accessed by pulling down the **Define** menu option from the main menu, and selecting **Packet Rate Matrix**. From the Packet Rate Matrix dialog window you can add packet flow information from source to destination. The packet flow information entered would not appear on the model, but when the model is run, the packet flow information would be generated.

## 4.94 Traffic Source: Response

### *Purpose:*

A Response Source is a message generator which is used to send messages in reply to received messages.

Packet routing for Response Sources is based upon the routing protocol for triggering messages. If the triggering message is part of a session and connection-oriented routing for sessions is specified, then the response packets will follow the session route. Otherwise they will be routed on a datagram basis.

Response sources are useful for modeling transaction responses, database replies, e-mail replies, etc. They are separate from ACK packets which are generated by the transport protocol's end-to-end flow control algorithm.

A Response Source may be created and represented directly on the user interface. Beneath the surface, the Response Source is translated into an application which executes a single answer message command. In other words, creating a Response Source is the same as:

- Defining an answer message command on a node
- Creating an application and attaching it to the node
- Calling the answer message command from the application.

Directly creating and editing a Response Source is easier than this three stage process, because it is not necessary to define a separate answer command on the node.

### *Creating:*

To create a Response Source pick the Response tool from the palette and drag a new response icon onto the background. Or use the create menu option to create a new response icon. Once a response icon is drawn on the background it must be connected to a node using one of the arc tools. Once created and connected, the response icon can be double clicked to bring up the details of the response message.

### *Connectivity:*

A Response Source can be connected to a Computer & Communications node, and to a Computer Group node. If connected to a Computer Group, then internally to the model there is a Response Source for each instance node.

The destination node for a response source message is the origin node of the incoming message that triggered the response.

### *Editing:*

Either double click on the Response Source icon, or highlight the icon and select the Edit/Detail menu option. The dialog box for editing the Response Source detail is then presented.

## 4. Modeling Constructs

The screenshot shows the 'Response Source' dialog box with the following configuration:

- Name: Resp16
- Icon: icn.response
- Priority: 1
- Routing class: Standard
- Trans protocol: Generic
- Packetize (ms): 0.0
- Msg size calc: Probability distribution
- Prob distrib: 1000.0
- Msg size units: Bytes
- Msg text option: Use original message
- Use \*ECHO\* in reports:

**Response Source Dialog Box**

***Execution:***

A Response Source is executed as an application with one answer message command.

For scheduling logic, see Source: Application.  
Only received message scheduling is used.

For packet generation and transmission, see Command: Answer Message.

***Reporting:***

- Messages: Delays For Message & Response Sources
- Packets: Statistics For Message & Response Sources

***Fields:***

See “Traffic Source: Application” and “Command: Answer Message” for field descriptions.

## 4.95 Traffic Source: Session

### *Purpose:*

A Session Source generates sessions; each session generates a sequence of messages. Sessions are useful for modelling switched or permanent virtual circuits or modeling a bursty message arrival process.

A Session Source may be created and represented directly on the user interface. Beneath the surface, the Session Source is translated into an application which executes a single session setup command. In other words, creating a Session Source is the same as:

- Defining a session setup command on a node
- Creating an application and attaching it to the node
- Calling the session setup command from the application.

Directly creating and editing a Session Source is easier than this three stage process, because it is not necessary to define a separate setup command on the node.

### *Creating:*

To create a Session Source, pick the Session source tool from the palette and drag a new session icon onto the background. Or use the create menu option to create a new session icon. Once a session icon is drawn on the background it must be connected to a node using one of the arc tools. Once created and connected, the session icon can be double clicked to bring up the details of the session.

### *Connectivity:*

A Session Source can be connected to a Computer & Communications node and to a Computer Group node. If connected to a Computer Group, then internally to the model there is a Session Source for each instance node.

### *Editing:*

Either double click on the Session Source icon, or highlight the icon and select the **Edit/Detail** menu option. The dialog box for editing the Session Source detail is then presented.

## 4. Modeling Constructs

**Session Source Dialog Box**

**Execution:**

A Session Source is executed as an application with one setup session command.

For scheduling logic, see *Source: Application*.

Delay Time scheduling is not supported.

The “Max Arrivals” field is not available.

For packet generation and transmission, see *Command: Setup Session*.

**Reporting:**

- Sessions: Setup Delay By Session Source
- Sessions: By Node
- Sessions: By Link
- Sessions: Length by Session Source
- Sessions: Blocking by Session Source

**Fields:**

See “*Traffic Source: Application*” and “*Command: Setup Session*” for field descriptions.



## 4.96 Transit Networks

### *Purpose:*

A transit network is an intermediate network that interconnects a collection of nodes. It provides an additional level of segmentation and reassembly at the boundaries of the transit net. It is called a transit network because its primary purpose is to model packets that enter on one side of the net, flow through the net, and finally exit on the other side of the net. In other words, it models packets transiting a network.

### *Link Behavior*

A transit net behaves as both a link and a subnet. From the external perspective of the nodes connected to a transit net, the transit net looks like a link that sends packets from the output buffer of one node connected to the net to the input buffer of another node connected to the net. The nodes connected to the transit net have no knowledge of how the packets are routed internally through the net to reach the desired node on the other side of the net.

### *Subnet Behavior*

Internally the transit net behaves like a subnet. It contains a topology of interconnected nodes, links, and traffic sources. It also has its own routing protocol. When a node has a packet to send across a transit net, the packet is segmented into new packets at the boundary of the net. The spawned packets are routed through the transit net to the target node on the other side of the transit net, where the original packet is reassembled. After reassembly, the original packet continues on its journey to its ultimate destination.

### *Protocol Layering*

The transit net introduces an additional protocol layer at the transit net boundaries. Packets are segmented and reassembled and flow from entry node to exit node in conformance with the protocol parameters for the type of connection between entry node and exit node. The segmented units that flow through the transit net can model packets, frames, cells, or any other protocol data unit. Since transit nets can be nested within transit nets, it is possible to model any number of protocol layers.

*TCP/IP over Frame Relay or ATM* For example, in a model where messages flow from source to destination under the control of TCP/IP, the TCP/IP packets could traverse a frame relay transit net on the way to their destination. When a TCP/IP packet arrives at the interface to the frame relay net, it would be segmented into frames, which would be routed through the net and reassembled into TCP/IP packets upon exiting the frame relay net. It is also possible that an ATM transit net could be embedded inside the frame relay transit net. When a frame arrives at the boundary of the ATM transit net, it would be segmented into cells, which would be routed through the ATM net and reassembled into frames upon exiting the ATM transit net.

### *Link-Level Flow Control*

One can also use the protocol layering capability of the transit net to do more detailed link-layer protocol modeling. For example, to model link-level flow control on a point-to-point link, one could define a transit net that contains only a single point-to-point link. The transit net would use a protocol with one of the windowed flow-control options for each connection across the “net” (which consists here of only a single link).

### *Compression or Encryption*

When a packet needs to be routed across a transit net, the packet is first converted to a data unit whose size is a linear function of the original packet. This linear transformation of the original packet size can be used to model compression or encryption of the packet prior to segmenting it for transmission through the tran-

## 4. Modeling Constructs

sit net. When the transformed data unit is reassembled at the exit node of the transit net, then a packet of the original size is delivered to the input buffer of the receiving node.

***Service Classes and Connection Types*** When a packet arrives at an interface to a transit net, the packet is mapped to a service class. Each port connected to a transit net has a list of service classes. A packet has a required network service class level (a number between 1 and 100). Each service class has a minimum service level and a maximum service level. A packet is mapped to that service level for which its required service level is in the interval [min svc level, max svc level]. The service classes defined at a transit net interface must serve non-overlapping service level intervals.

Each service class has a list of connection types. Each connection type has list of specific destination nodes on the boundary of the subnet. A separate connection instance is created during the simulation for each of the nodes in the connection type's destination list. Each connection type also has a priority, routing class, protocol parameters, compression factor, segmentation time, and a few other parameters. A service class includes a DEFAULT Connection Type with no explicitly listed destinations.

After a packet has been mapped to a service class, it is assigned to the connection instance for the exiting node on the other side of the transit net. For a given service class, a destination node can appear only once among the class's list of connection types. If a connection to the desired node does not exist, the DEFAULT Connection Type for the service class is used.

By using multiple service classes, different classes of packets can be transmitted to the same destination using different protocol parameters. For example, in an ATM transit net, both ABR and CBR traffic may need to be transmitted between the same pair of switches.

By using different connection types for a given service class, deliveries to different destination nodes can be controlled by different protocol parameters. For example, in a frame relay network, the CIR to one node may be 128 kbps and another node might be 256 kbps.

When a node is initially connected to a transit net, the resulting port's initial list of service classes is copied from the list of service classes included in the transit net's parameter set. In the transit net's parameter set, none of the service class connection types have assigned destinations. The destination nodes for each connection type are assigned only at the port interfaces to the transit net. The connection types defined in the transit net's parameters serve only as templates for initializing the connection types at individual ports.

### ***Interface Ports***

Arcs connect nodes to access points on transit network boundaries. The access point graphically represents the connection to a link inside the transit network. The arc connecting to a transit net access point represents the node's port to the transit net. The port's input and output buffers store incoming and outgoing packets as usual. The arc also has attributes for an additional "interface" port. The interface port has an input and output buffer that is used by the segmented packets. The interface port buffers can have different attributes from the regular port buffers.

***Traffic Constraints***

Only packet traffic can flow through a transit net. At the present time, call traffic cannot be routed through a transit net. A transit net can include message, response, and session sources. It can not include call and application sources.

The destinations for messages originating inside a transit network are limited to nodes in the same transit network. In addition, messages originating outside of a transit net can not terminate inside a transit net. As a consequence of these constraints, when a transit net is cloned, any nodes in a traffic source's destination list are changed to the cloned nodes in the cloned net instead of pointing back to the nodes in the original net.

The primary purpose of the traffic sources inside a transit net is to model the net's own routing or administrative traffic. The model assumes that the other traffic carried by the net is flowing through or transiting the net.

***Session Routing***

Transit nets do not internally enforce connection-oriented routing of sessions that transit the net. Thus, packets flowing as part of a connection-oriented session will always enter and leave a transit net at the same pair of nodes, but inside the transit net the segmented packets are routed independently. To force segmented packets to follow the same routes inside of transit nets, avoid adaptive or multipath routing options.

***Multicasting***

When a multicast packet arrives at the boundary of a transit net, a separate copy of the packet is cloned for delivery to each of its ultimate multicast destinations. Thus, each packet routed across a transit net is a single destination packet. Within a transit net, multicasting operates as always in COMNET, where the destination list is partitioned among selected outgoing ports and one copy of the packet is cloned for each selected port, rather than each ultimate destination.

## 4. Modeling Constructs

### 4.97 Transport Protocol

***Purpose:***

Transport Protocol is a label which is applied to a message source. It carries with it a description of the end-to-end network parameters which need to be applied to the data flow between the origin and destination.

Different Transport Protocol labels may be applied to different message sources. As many Transport Protocols as desired by may be defined in the model.

COMNET III includes parameters for TCP/IP, UDP/IP, NCP/IPX and NCP/IPX Burst Mode in the Transport Protocol Library. However, other models of TCP and UDP may be added as well as models of all other kinds of transports.

***Creating:***

To create a Transport Protocol, edit the appropriate type of traffic source (message, response, session source, or equivalent command) and click on the double dot button at the side of the transport protocol box. Also the **Define/Transport Protocols** menu option may be used to create and/or edit transport protocol definitions.

A Transport Protocol definition may also be added to the system library via the **Archive/Objects** menu option. If a Transport Protocol is defined in the system library it will become available when a new model is made, or when the system library definition is specifically copied into the local model.

COMNET III always creates a Transport Protocol class called "Generic" when a new model is made.

***Connectivity:***

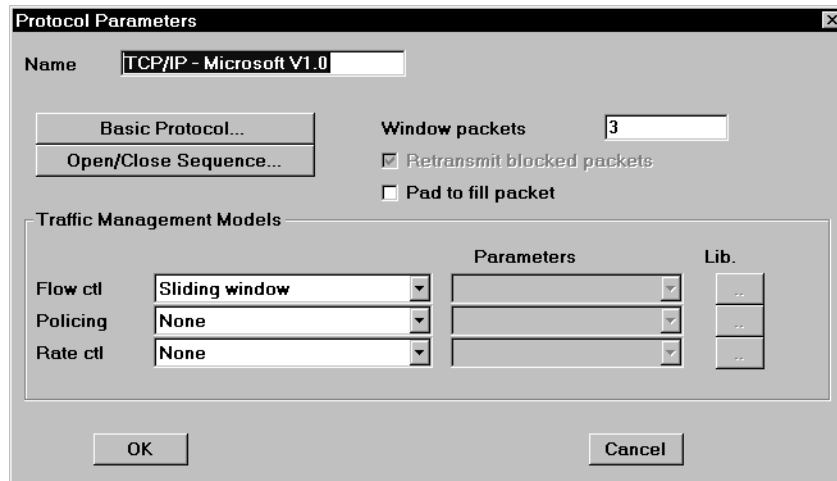
A message, response, and session source and a transport, answer, and setup command each references a transport protocol in the model's list of transport protocols.

Call traffic does not have a transport protocol.

***Editing:***

Double click on a message source icon to bring up the source dialog box, or highlight the icon and pick the **Edit/Detail** menu option. The Transport Protocol to use for the source can then be applied in the Transport Protocol value box.

The Transport Protocol parameters themselves are edited by clicking on the double dots next to the Transport Protocol value box, and then by selecting the **Edit** button.



Transport Protocol Dialog Box

**Execution:** See field descriptions below.

**Reporting:** N/A

### Fields:

**Name** The name of the transport protocol.

**Packet Data Bytes** When a message is created, the size of packet to use. On message dialog boxes the total size of the message is specified in terms of bytes or packets.

For instance, if a message size of 1024 bytes were used and the “Packet Data Bytes” were set at 128 bytes, then 8 packets would be generated for the message. When a message is sized in bytes, it is possible that the last packet in the message is not full—COMNET III does not pad the last packet to full size.

Alternatively, if the message size was set to 8 packets and the “Packet Data Bytes” was set to 128, then the same amount of data is created. When a message is sized in packets, the requirement is for so many full packets—there are no partially full packets.

**Packet Overhead Bytes** The number of bytes added by the transport protocol which will be carried between the origin and the destination. This would reflect routing information, CRC checks, etc., that are considered as part of the end to end data that has to be sent. Such data has to be switched and buffered at each node, and transmitted over each link.

The number of packet overhead bytes is added to the packet data bytes to determine the total packet size.

This is different from the framing overheads that are specified on each link. The frame overheads are used to model the link layer (i.e., layer 2) overheads and are included in the transmission time calculations in addition to the “Packet Over-

## 4. Modeling Constructs

head Bytes.” However, the frame overheads are not buffered or switched by the intermediate nodes.

### Window Size

The end-to-end window size to be used in association with the flow control algorithm.

An application with a transport or answer command that is waiting for flow control authorization is not eligible to execute until an ACK is received.

For instance, if fixed window flow control is in use and the window size is set to 1, then every packet arriving at the destination will cause an ACK to be sent back to the origin. The origin cannot create and send the next packet until the ACK from the preceding packet has been received back at the origin.

Note that the flow control is end-to-end. COMNET III does not model link layer (layer 2) flow control.

### End-to-End ACK Bytes

The size of the ACK packets generated by end-to-end flow control.

### Retransmit Time

If a packet is blocked en route to its destination, it will be retransmitted after the “Retransmit Time” from the origin.

A packet may be blocked because some port input buffer or port output buffer has insufficient space to receive it, some node has insufficient total input or total output buffer space to receive it, or a routing decision at some point on the route cannot route the packet further. It may also be blocked if one or more frames carrying part of it was discarded by the WAN cloud.

Note that a packet blocked because the selected port output buffer, or the total output buffer space, on an ATM node has insufficient space will cause the packet to remain in the port input buffer on the ATM node. In this case the packet will not be retransmitted from the origin.

If the option to retransmit blocked packets is inactive, then blocked packets are dropped. The blocked packet will be counted as dropped and then discarded. Any message generating such a packet will not count as being delivered

The retransmission timer in Release 1.2 is improved to account for the age of the packet before it is blocked. For the sake of simulation efficiency, COMNET does not model individual timers on each packet to determine whether a packet needs to be retransmitted. Instead, it schedules a packet retransmission only after the packet is actually blocked somewhere in the network. In earlier releases, the specified retransmission time was applied after the packet was blocked (under the assumption that the packet delay through the network was insignificant compared with the retransmission time). In release 1.2, the retransmission time is adjusted so that the packet is retransmitted the specified delay after the packet was originally transmitted. The option is available for backward compatibility.

Note that because COMNET III doesn't model the retransmission timers explicitly, COMNET will not model the retransmission of a packet that is late but still in transit through the network. COMNET III provides an acknowledgement delay report to flag when this condition is happening in the model.

Setting the “Retransmit Time” to 0 and having a routing decision at the origin return no route will effectively put the model into an infinite loop as a packet is blocked and then immediately reattempted. Simulation time may advance if non zero packet switching times have been entered on the originating node, or a packetizing delay has been specified.

### **ACK Priority**

ACK packets are routed back to the origin as if they were data packets. It may be that ACK packets should have higher priority at intermediate nodes in order that windowing delays can be minimized. If so then the “ACK priority” can be increased.

The result of increasing priority is to place the packet at the head of the port input buffer and port output buffer queues on the nodes where the packet is switched.

### **Error Control Ack**

Used for modeling an extra acknowledgement that acknowledges the delivery of the packet or window at the destination but does not allow the window to be advanced. This can be considered an error control acknowledgement that may appear as a separate acknowledgement when flow control is used, or as an acknowledgement when no flow control is used. COMNET III's error control mechanisms do not require the explicit modeling of these acknowledgements so that this option is primarily to model the traffic introduced by their presence. These extra acknowledgements impact local area networks because their timing is slightly delayed from the delivery of the packet and thus enhances the potential for collisions on random access links (such as CSMA/CD) and uses a token on the token-ring networks.

### **Negotiated Packet and Window Sizes**

Allows the protocol's packet size and window size may be reduced by constraints on the socket to which the source is attached and the socket to which the message is destined. This refinement is determined at the start of each new message so that different messages from the same source may have different packet and window sizes when they have different destinations.

### **Padded Packets**

A way to model protocols that are constrained to always transmit full sized packets (such as ATM). This padding option is only available for the enhanced protocols or for when no packet retransmission is required. When data are available for transmission but these data do not fill a full-sized packet, this option allows COMNET to model a full-sized packet even though the delivered data is less than the full packet.

**Retransmit Blocked Packets** Indicates whether blocked packets should be retransmitted or dropped.

### **Flow Control Method**

None  
 Fixed Window  
 Sliding Window  
 SNA Pacing Control  
*(see appropriate Flow Control entry)*

## 4. Modeling Constructs

### 4.98 Trigger Events

*Purpose:*

Triggers can be used to model time dependencies among traffic sources, and also time dependencies between traffic sources and significant state changes of nodes and links. An individual trigger specifies that, when the associated traffic source or node or link enters a specified state, a specified successor source is scheduled. For example, using triggers you can easily model the situation in which a multi-cast routing table message is sent from a central location a short time after a node or link fails, or the situation in which call traffic is initiated in response to the arrival of a packet-switched message, or vice-versa.

Successor sources can be scheduled by trigger only, meaning they execute only when a trigger causes them to execute, or they may be scheduled by any other means. In the latter case, executions caused by triggers will be in addition to executions caused by other means. For example, if you have a message source, say Msg1, scheduled by iteration time, say every 1 minute, and you also have a session source, Ses1, that triggers Msg1, then Msg1 will execute every minute and also every time Ses1's trigger fires.

It is important to keep clear the distinction between a *triggering* source/node/link, and a "triggered" source, referred to as the *successor*. The trigger itself will be found in a list accessed by clicking the Triggers button on a source/node/link. The source/node/link whose trigger list you are examining, say Msg1 for example, is the triggering source/node/link. Another way to refer to the triggers in the list is by ownership. We say the triggers in the list are Msg1's triggers. Msg1's triggers will fire when Msg1 enters a specified state.

When you edit a trigger in the list, you will see that the trigger has a number of attributes, one of which is the successor. That successor will be scheduled when the conditions of the trigger are met. When the successor is scheduled, we say the trigger *fires*.

The conditions of the trigger, referred to as the *triggering rule*, are chosen from a set that varies depending on the triggering source/node/link. For example, for a message source, say Msg1, the list of rules includes "upon source startup," which means the trigger will fire when Msg1 starts executing. Msg1's list also includes "upon message creation," which means the trigger will fire when Msg1's message is first created. On the other hand, a link will have a very different set, including "upon alarm first sounding" and "upon link failure."

Other attributes of the trigger include the *triggering delay*, which is the delay that will elapse between the firing of the trigger and the actual execution of the successor. In addition, depending on the triggering rule, you might have to specify the command or alarm to which the rule applies. For example, you may want an application source to have a trigger that fires "upon message creation," but the application creates several different messages, so you have to specify to which message the rule applies. This additional information is referred to as the *rule qualifier*.

When the trigger list contains more than one trigger, all of the triggers must have the same triggering rule and qualifier, so the only differences among them are the successors and the delays. Such a list is interpreted in one of two ways, as specified by the Weighted List check box: Either all of the successors in the list are triggered when the rule is satisfied, or one of the triggers is chosen randomly,



based on a sample from the uniform distribution and a probability (weight) assigned to each trigger.

Here is a summary of the possible trigger rules:

<b>Trigger rule</b>	<b>Explanation</b>
upon source startup	When source begins execution
upon source completion	When source completes execution
upon msg creation	When specified message is first created
upon msg delivery	When specified message is successfully delivered to all destinations
upon session initiation	When session setup begins
upon session establish	When session setup completes and session is established
upon session completion	When session is terminated
upon alarm first sounding	When specified alarm first sounds
upon alarm clear	When specified alarm stops sounding
upon link failure	When link fails
upon link recovery	When link recovers after failure
upon node failure	When node fails
upon node recovery	When node recovers after failure
upon filter time-out	When specified filter command's timer expires

#### 4. Modeling Constructs

## 5. Creating COMNET III Models

### 5.1 Introduction

COMNET III's graphical user interface is used to create and modify a network description. The network description is saved in a file (modelname.c3) for future COMNET III sessions so it can be modified or run with new operating parameters.

This chapter describes COMNET's graphical user interface and shows how to use it to create a communications network model. It focuses on how to build a model rather than the details of the model itself. Previous chapters provided detailed background on the underlying network model components and their attributes.

This chapter uses some screen shots of COMNET III menus and dialog boxes to illustrate important features, but it is not an exhaustive tour of the user interface. Since graphical user interfaces are, by their nature, designed to be intuitive, it isn't necessary to explain the operation of each dialog box.

To start COMNET III, double-click on the COMNET III icon or type the command **comnet3**, depending on the computer and window system you are using. The COMNET III logo is displayed while COMNET III initializes. After initialization is complete, a tool palette appears in a column along the left edge of the display and a menu bar with pull-down menus appears across the top of the display.

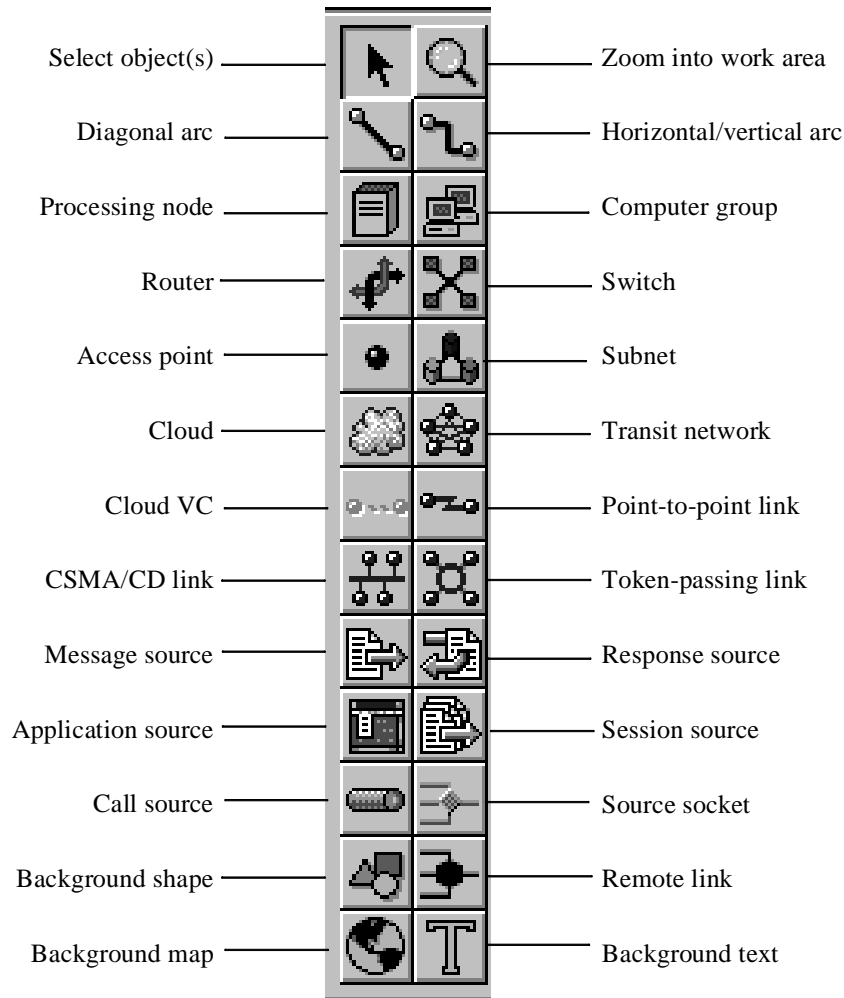
COMNET III's main window is shown in Fig 4.1. It is used to build and edit models, run animated models and to show model results.

### 5.2 Using the COMNET III Tool Palette

The tool palette on the left side of the COMNET III display includes tools for creating Nodes, Links, Traffic Sources and tools for editing.

Most of the tools in the palette can be activated in two ways. If you click once on a tool it becomes activated. The tool which has been selected changes appearance so that it looks as if it has been pushed in while the remaining tools continue to appear raised. This is **standard mode**. Having activated the tool, you can carry out one action. As soon as you complete the action, the chosen tool becomes inactive and the selection tool takes its place as the active tool in the palette. You can then use the selection tool or return to the palette to activate another tool.

## 5. Creating COMNET III Models



**COMNET III Tool Palette**

If you double click on a tool, it will stay activated and allow multiple repetitions of an action until you return to the palette to activate another tool. This is extended mode. Tools which have been activated this way take on a more deeply pushed in appearance.

### 5.2.1 Object Creation Tools

To create a node using the palette, select the desired node tool on the palette by clicking on it, then, while holding down the mouse button, drag the tool outline to the desired position on the screen then release the mouse button. An alternate technique is to click once on the desired node tool, move the cursor to the desired position, then click again.

Links are created in the same manner as Nodes.

Traffic sources are created in the same manner as Nodes.

To create multiple objects without having to repeatedly select the object's tool from the palette, you can click twice on the object's tool icon in the palette to place it in extended mode. Then, each time you click on the display, another object is created. Thus, after creating some nodes, you can click twice on a link tool to place it in extended mode. This will "lock-on" the link-create mode. Then every time you click on the display, another link is created. Finally,

to connect the nodes to links, click twice on one of the palette's connection tools to “lock-on” the connect mode. Once you are in connect-mode, you do not need to click repeatedly on the connect tool.

To return to the standard mode of operation, click on the selection tool, which is the arrow at the top left of the palette.

Nodes, links and traffic sources are created with default attributes, including default ID codes and default icons. To change the attributes of an object, select the object by clicking on it. Then click on Edit (on the menu bar) and then Detail (on the pull-down menu). Edit/Detail displays a dialog box showing the current attributes of the object. A simpler technique is to simply double click on the object's icon to bring up the Edit/Detail dialog box.

A helpful technique for keeping track of which nodes and links have had their attributes set to the values you desire and which ones have only default attributes is to look at the ID code displayed next to the icon. Unique default node, link and source ID codes are automatically assigned by COMNET III when objects are created. Examples of default IDs are **Link10**, **Node14** and **Msg25**. All types of Nodes are labeled **Node**. All types of Links are labeled **Link**. Traffic sources are labeled by type of source: **App**, **Msg**, **Call**, **Sess**. Subnetworks are labeled Net. Network Pads are not labeled. When you edit the detailed attributes for an object, it is a good practice to specify your own ID code so you can recognize at a glance which objects have been edited and which ones have not.

### 5.2.2 Connection Tools

To connect a traffic source to a node or a node to a link, select one of the connection tools by clicking on it. Then click once on each network object to be connected. The order does not matter. The connection is represented by a line between the elements.

Alternatively, you can click on the first object, move to the second object while holding down the mouse button then release the button when you have arrived at the second object.

There are two connection tools. The standard “jointed arc” tool represents the connection with a direct line between the network elements using the shortest path. If the user desires, the path can be represented by a sequence of line segments drawn at the users discretion. Click once on the first object, click once on each intermediate point desired, then click on the second object. Multisegment connections have no effect on the function of the connection.

The other tool represents the connection with lines which travel only at right angles. As with the standard connection tool, a sequence of intermediate points can be specified, but the tool will ensure that all lines drawn to connect the points are at right angles.

There is no functional difference between the tools, the distinction is purely aesthetic.

If a connection does not make sense, nothing will happen. Connections are only valid between nodes and links and between traffic sources and nodes. In a backbone layout, connections are also valid between links and network access points. In a subnetwork layout, connections are valid between nodes and network access points.

If you are not happy with the resulting appearance of the connection, simply delete the connection (select it, then choose Edit/Cut) then redraw it with greater artistry.

Note that only two nodes can be connected to a point-to-point link. This restriction is enforced by the layout editor. For multinode links, you should choose CSMA/CD, CSMA, Token Passing, Polling or ALOHA links instead. It is also possible to change the link access protocol on an existing link by selecting the link and using Edit/Detail.

### 5.2.3 Selection Tool

Objects must be selected before performing actions such as edits. To select an object, first ensure that the selection tool has been activated. This is the arrow at the top left position of the tool palette. Once the selection tool is acti-

## 5. Creating COMNET III Models

vated, you can select an object by clicking on it once. It will turn red to indicate that it has been selected. Once an object has been selected, use **Edit** on the menu bar and one of the functions on the **Edit** pull-down menu (**Cut**, **Copy**, **Select All**, **Detail...**, **Move**, **Enter**, **Leave**, **Paste**, **Clear**, **Duplicate**, **Clone...**, and **Scale**) to manipulate the object. Double clicking on an object is a short cut for performing an **Edit/Detail** after selecting the object.

### 5.2.4 Selecting a Group of Objects

The extended select mode lets you select a group of objects on the display and then perform operations on all objects in the group. To select multiple objects, click twice on the selection tool displayed at the top left of the palette. This places the selection tool into extended select mode.

Then each time you click on an object, it is added to the group of selected objects. Each selected object turns red to indicate that it has been added to the group of selected objects. When the selection tool is in extended mode, objects on which you click will be toggled in and out of select mode each time you click.

Once in extended select mode, you can also select a group of objects by drawing a selection rectangle around them. This is accomplished by clicking at one corner of the desired area then dragging to the opposite corner of the desired area while holding the mouse button down. When the desired area is surrounded by the rectangle, release the mouse button. All objects included in the rectangle will be selected.

To select all items in a model, choose **Edit/Select All**.

When you perform **Edit/Detail** on a group of objects, COMNET III automatically cycles through the dialog boxes for each selected object as you click on **Ok** or **Cancel**.

### 5.2.5 Text Tool

The text tool is used to add text labels anywhere on your model layout. Select the text tool icon on the palette then click on the position where the text is to be placed. A dialog box appears. Type the text into the dialog box then select **OK** or **Cancel**.

Once the text appears, it can easily be moved to adjust its position and its size can be changed using **Edit/Scale**.

### 5.2.6 Background Icon Tool

Often it is useful to place the model layout against a background which adds information to the layout and improves its appearance. A typical use might be to place a model of a Wide Area Network against a map.

The background icon tool allows the user to insert into the layout any icon from the “user.sg2” graphics library file. The default “user.sg2” file already contains several pre-defined maps. This manual's appendix on the graphics editor explains how to create or import additional icons into that library file. The imported graphics can be simple line drawings or photographs stored as bit maps.

To insert a background icon from the “user.sg2” file, choose the background icon tool on the palette then click on the layout. A dialog box appears. The dialog box contains the names of each icon in the “user.sg2” file. Select the desired icon then click **Ok** or **Cancel**. Once the icon appears, it can be moved and scaled if desired.

### 5.2.7 Subnetwork Node Tool

COMNET III allows subnetworks to be represented as icons within a larger network. This allows for higher level abstractions of a network to remain uncluttered, but makes the underlying detail easily available for inspection and editing.

This alternative representation of the layout does not cause any shortcuts to be taken in the model's operation. It simply allows an uncluttered and hierarchical model layout. When the simulation is run, all components of the sub-network such as nodes, links and traffic generators fully participate in the simulation as if they were in the main layout.

The subnetwork node tool is selected from the palette and inserted into a model layout just as any other node would be. When this subnetwork icon is selected, the screen switches to a representation of the subnetwork's layout screen and the model name which normally appears at the top left of the screen is supplemented with the subnetwork's name. Thus the title would change to **ModelName: SubnetName**.

A subnet may now contain other subnets, and there is no limit to the nesting depth. However, nested subnets differ from non-nested subnets in one very important respect: a nested subnet does not have its own routing domain. Instead, routing takes place as if there were no nested subnets. In other words, the highest subnet of the nest establishes a routing domain that includes all the nodes contained therein, whether or not those nodes are contained within nested subnets.

Nested subnets are really only for graphical convenience; they reduce clutter in the model layout, but have no effect on simulation.

Connections between subnetworks and the main network are accomplished through access points which appear at both levels of the model. The access point tool on the palette is grayed out and deactivated when at the main level. Access points can only be inserted in the subnetwork. They will then show as connection points to the subnetwork icon when viewed from the main level.

To switch from the main network layout to a sub-network, double click on a sub-network icon.

To switch back to the main network layout from a sub-network, double click on the background of the sub-network.

## 5.3 Moving or Repositioning Objects

You can move individual objects (node, link, traffic, text or background icons) to another position on the display by clicking once on the object, holding the mouse button down while dragging to the new location then releasing the mouse button at the desired location. Connections to the object adjust automatically after the moves.

To move a group of objects, select several objects using extended select mode. Select **Edit** then **Move** from the menu. A rectangle containing the objects appears. Drag the rectangle to a new location then click to anchor the objects in their new position.

## 5.4 Moving Around the Layout

If you have a large model or are zoomed in, its layout may not fit in the viewing window. You can either pan around the model, viewing portions of the model through the viewing window, or you zoom out to see the entire model layout at once.

Panning is accomplished by using the vertical and horizontal scroll bars along the right side and bottom of the screen. These operate identically to those in other applications on your machine. Note that the size of the scroll button indicates how much of the model you are seeing at any given time. If the button fills the entire scroll bar, you are seeing all of the model (on that axis). If the button fills half of the scroll bar, you are seeing half of the model.

## 5.5 COMNET III Menus

Just as the tool palette gives the user access to the more frequently used parts of COMNET III graphical interface, the COMNET main menu bar and its pull down menus give the user access to the finer details of COMNET.

Each item on the main menu bar has a pull down menu with a number of choices. Items in pull down menus which are followed by an ellipsis will pop up a dialog box when selected. Some dialog boxes contain choices which yield pull down lists or yet more dialog boxes.

## 5. Creating COMNET III Models

### 5.5.1 File Menu

<b>N</b> ew	Ctrl+N
<b>O</b> pen...	Ctrl+O
<b>S</b> ave	Ctrl+S
<b>S</b> ave As...	
<b>I</b> mport	▶
<b>E</b> xport	▶
<b>M</b> erge...	
<b>P</b> rint...	Ctrl+P
<b>E</b> xit	

The File menu offers standard functions the user will probably be familiar with. Any action which might result in loss of information will cause COMNET to prompt you to save the existing model. Thus if you quit, open another model or create a new model, you will be asked if you want to save the current model.

- New** Clears the current model, if any, and creates a “clean slate” on which a new model can be built.
- Open...** Brings up a dialog box to open an existing model. The user specifies the path and the directory name (without the “.C3” extension), or uses the mouse to navigate through the file system to find the model.
- Save** Saves the current model using the name shown in the upper left of the screen just above the tool palette.
- Save As...** Saves the current model using a path and directory name specified by the user.
- Import** Imports either a bitmap, a COMNET III external model file format model, or a network topology file generated by third party software. Supported third party software includes Cabletron Spectrum Topology files, HP OpenView topology files, IBM NetView 6000 topology files, Castlerock SNMPc topology files, and DEC PolyCenter topology files. Once a model has been saved a network traffic profile can be imported and attached to the model. Supported third party software profiles include Axon Networks, Inc. LANServant RMON II profile, Network General’s Expert Sniffer Profile, and Frontier’s NETscout RMON II probe profile.
- Importing NMS Topology Files**  
Please consult the COMNET Baseline User’s Guide to properly import your NMS topology file.
- Importing Network Traffic Files**  
Please consult the COMNET Baseline User’s Guide to properly import your network traffic files.
- Export...** The Export function saves the model file in ASCII or COMNET III external model file format (\*.c3e), rather than binary format. If you are moving from one computer platform to another you need to convert the model file to ASCII before moving it. Note that the Load function will accept binary or ASCII model files. Binary files load faster than ASCII files.



You can also export the simulation statistics to a file that can be loaded into most spreadsheets, such as Microsoft Excel. See Simulation Statistics in Chapter 4 for more information.

You can also export an image of the model layout to either an encapsulated PostScript file (\*.eps) or to a bitmap.

### **Merge...**

Merges the current model file with specifications from a COMNET III external file format model (\*.c3e). If you are familiar with the external file format, this may be the fastest way to test a number of models with only small variations in each one. For more information see the COMNET III external model file format document.

### **Print...**

Prints a copy of the current layout screen to an attached printer

### **Exit**

Exits COMNET III. Prompts the user to save the model if necessary.

## 5. Creating COMNET III Models

### 5.5.2 Edit Menu

<b>C</b> ut	Ctrl+X
<b>C</b> opy	Ctrl+C
<b>P</b> aste	Ctrl+V
<b>D</b> uplicate	
<b>C</b> lear	Del
<b>C</b> lone...	
<b>S</b> elect <b>A</b> ll	Ctrl+A
<b>S</b> elect By List...	
<b>F</b> ind...	Ctrl+F
<b>P</b> arent...	
<b>P</b> roperties...	Alt+Enter

Most Edit menu commands will be familiar to users of windowed user interfaces.

- Cut** Deletes the selected object (or objects) and holds a copy in the clipboard so it can later be used in a **Paste** operation. Any object previously in the clipboard is replaced.
- Copy** Places a copy of the selected object (or objects) in the clipboard so it can later be used in a **Paste** operation. Any object previously in the clipboard is replaced.
- Paste** Pastes a copy of the objects which have previously been placed into the clipboard at the location specified by the user.
- Duplicate** Provides the same functionality as a **Copy** followed immediately by a **Paste** of the selected object.
- Clear** Deletes the selected object or objects but does not place a copy in the clipboard.
- Clone...** Provides the same functionality as **Copy**, but it brings up a dialog box to ask how many copies to make, then it immediately inserts the copies next to the original. User can specify the x-y offset which will be used when the clones are inserted next to the original.
- Select all** Selects all objects in the model.
- Select by List...** Allows you to select any object or objects in an open model from a listing of all objects in the model.
- Find...** Allows you to search for an object in an open model by the object name, a parameter name or an attribute title. If a match is found, the object will be highlighted.
- Parent...** Allows you to edit the detail of the next level up without leaving the level you are at. If you are within a subnetwork or cloud you can edit the detail of the subnetwork or cloud. Choosing Edit/Parent at the top level lets you edit the backbone detail; it is the same as choosing Define/Backbone Properties... .
- Properties...** Brings up the dialog box of the selected object so that its parameters can be edited. When multiple objects are selected, their dialog boxes are presented for editing one at a time. This is particularly used to edit the routing protocol details for selected subnetworks, and the port details of arcs connected to links.

### 5.5.3 View Menu

<b>By List...</b>	
<b>Z</b> oom <b>I</b> n	
Z <u>o</u> om <b>O</b> ut	
<b>F</b> it to <b>W</b> indow	
<b>V</b> iew (1:1)	
<b>H</b> ide	
<b>S</b> how	
<b>S</b> how <b>A</b> ll	
<b>T</b> oggle <b>N</b> ames	
<b>T</b> oolbars	▶
<b>E</b> nter	Ctrl+E
<b>L</b> eave	Ctrl+L

The View menu controls how the elements of the model are presented on the COMNET III layout screen.

#### By List...

Presents a dialog box where you can access all the different elements of your model and edit their properties. Items in the list can be folders containing other items, in which case the label of the item is appended with “(+),” “(-),” or “(empty),” indicating that the folder is closed, open, or empty, respectively. Non-empty folders can be opened and closed using buttons or by double-clicking. The Open All button will open the selected folder, any folders contained in it, any folders contained in them, and so on.

At first, the list will be empty. You can add items to the list by clicking on one or more categories in the “include” group of check boxes. Clicking the All button is the same as clicking all the check boxes.

In addition to being able to control which main categories are included in the list, you can filter out specific information from the list. Filtering is handy for focusing in on specific items of information. You enable the filtering action by clicking the “filter out” check box. At first, a small default set of filtered items will be in effect. Any subsequent changes to this default set will be saved with the model, so items you routinely wished filtered will be remembered.

You can change the default filter set in several ways. You can filter out individual items by selecting an item in the list and clicking the Selected Item button. If the item is a group of objects, for example “Routers,” the whole group will be filtered out. If the item is a single object, that object will be filtered out. If the item is a property of an object, that property will be filtered out for all similar objects.

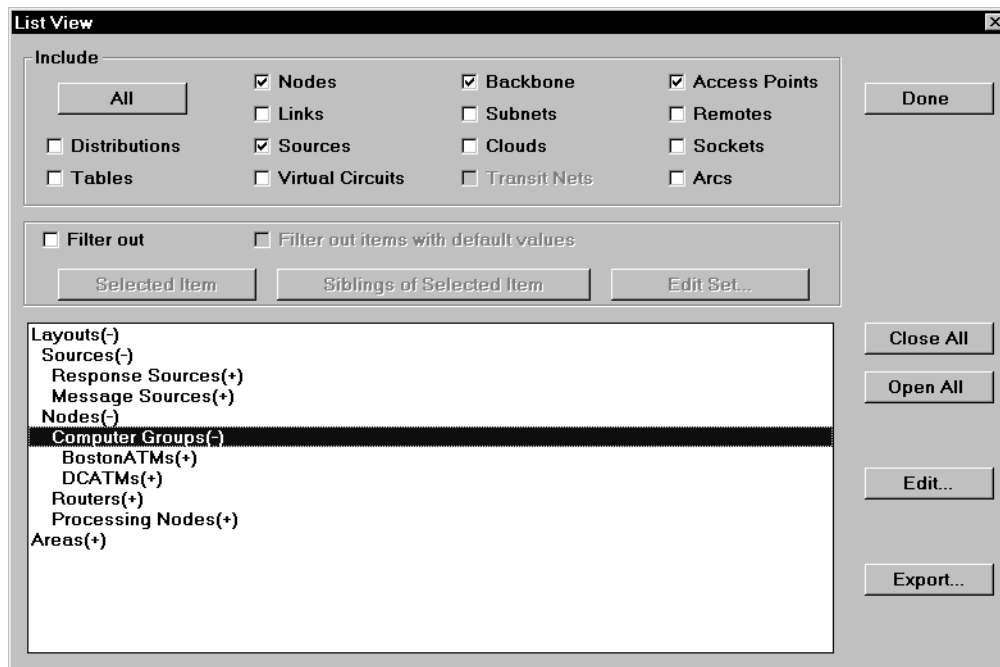
If you’re interested in only one item of information, you can hide all the siblings of that item by selecting it and clicking the Siblings of Selected Item button. If you filter too much and you want to “un-filter” an item, click the Edit Set button. You’ll see the filter set, and you can remove individual items from it. The “Filter out items with default values” checkbox, when checked, causes all library-based items that still have their default values to be temporarily filtered out.

## 5. Creating COMNET III Models

The Edit button brings up a Properties dialog for the object or objects selected. Multiple objects can be edited sequentially by selecting a folder containing more than one object.

The Export button exports the information to a text file. This feature is the same as that available via the Export to External Model File option of the File menu, with two additional controls:

1. You can select between either the External Model File (.C3E) format, or a straight echo of the hierarchical list.
2. You can choose whether to include the filtered data in the export (it is included by default). Note that by not including filtered data, you will obtain a partially complete model file, which may have trouble loading, so use this option carefully.



### Zoom In

After selecting this command, click once at the upper left of the area to be selected. Then drag the mouse to the lower right of the area you want to zoom in on. A rectangle appears as you drag indicating the proposed zoom area. Once the mouse button is released, the view zooms in to the selected area.

### Zoom Out

This command expands the view to take in the entire work area. If the work area had been expanded, the resulting view would look like the center or right diagram below.

### Fit to Window

If you have a model that has objects outside of the visible display area and you wish to see the whole model in the display area then use this command to see the entire model in the display area.

### View (1:1)

Allows you to hide a selected object or objects in an open model.

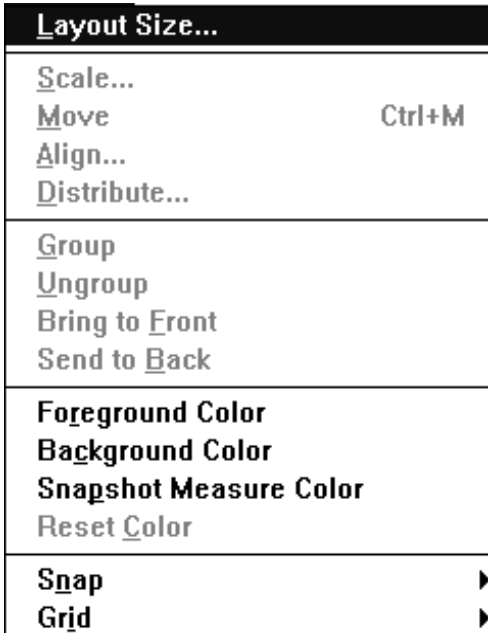
### Hide

Hides the selected item. You may wish to hide certain portions of your layout to

	decrease the screen complexity. They will remain in your model functionally.
<b>Show</b>	Displays an item that has been hidden.
<b>Show All</b>	Unhides all elements of the model that have been hidden.
<b>Toggle Names</b>	Turns names on and off. This can be useful because in a complicated model layout the names may overlap and cause confusion.
<b>Toolbars</b>	This menu item lets you turn the color palette, menu toolbar, and 3-D toolbar on or off. The toolbars may also be moved to anywhere on your screen.
<b>Enter</b>	If the selected object is a subnetwork, this will leave the top level of the layout and enter the subnetwork. When a group of objects are selected, the first selected subnetwork is entered. This item is grayed and disabled when in a subnetwork layout.
<b>Leave</b>	When in a subnetwork layout, returns to the main model layout. This item is grayed and disabled at the main layout level.

## 5. Creating COMNET III Models

### 5.5.4 Layout Menu



#### Layout Size...

Sets the size of the work area in terms of the number of “screens.” For example, if the view were set at 1:1 and the size is two screens, then there is twice as much work area as is visible.

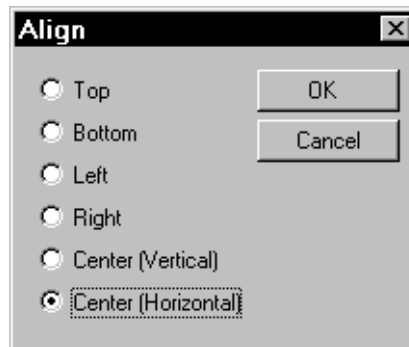
#### Scale...

Changes the size of the icons which represent objects on the layout. The user specifies the scaling factor. A scaling factor of 2 makes the icon twice as big, while a scaling factor of 0.5 makes it half the original size. This can be used to fine-tune the size of background icons.

#### Move

Allows a group of selected objects to be moved on the layout screen. Single objects can be moved simply by clicking on them, dragging to a new position and releasing the mouse button.

#### Align...



With this command you can align a number of selected objects according to various criteria.

Top will align the tops of the selected objects.

Bottom will align the bottoms of the selected objects.

Left will align the left sides of the selected objects

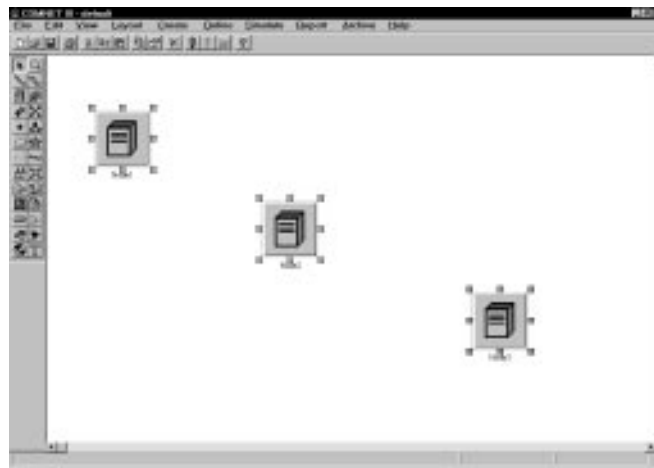
Right will align the right sides of the selected objects

Center( Vertical) will align the centers of the selected objects along a vertical line.

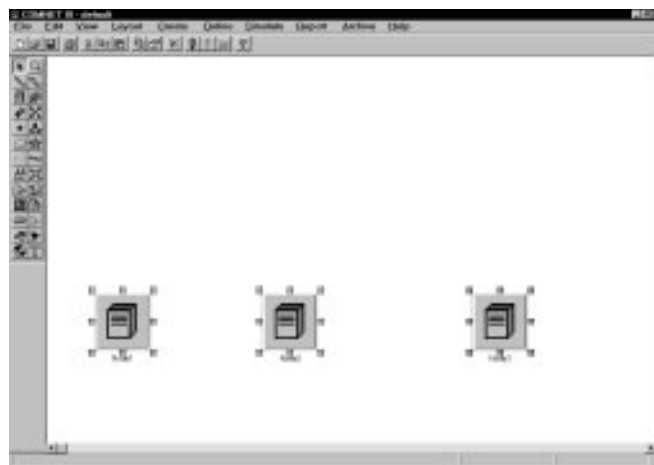
Center( Horizontal) will align the centers of the selected objects along a horizontal line.

The screen shots below show the effect of using Align.

Before aligning the the horizontal centers:

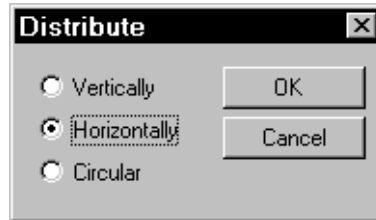


After aligning the horizontal centers:



## 5. Creating COMNET III Models

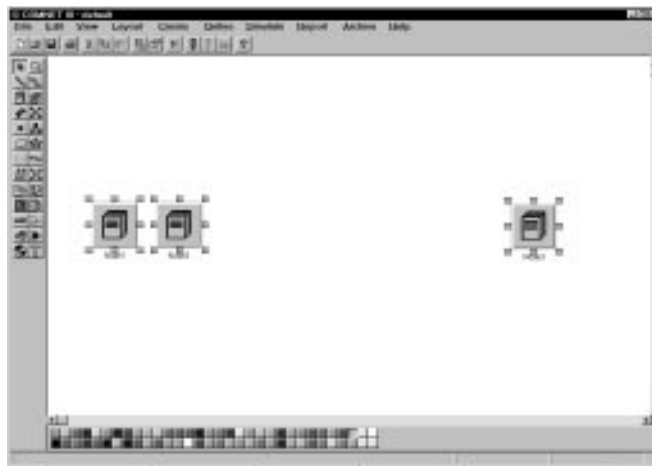
### Distribute...



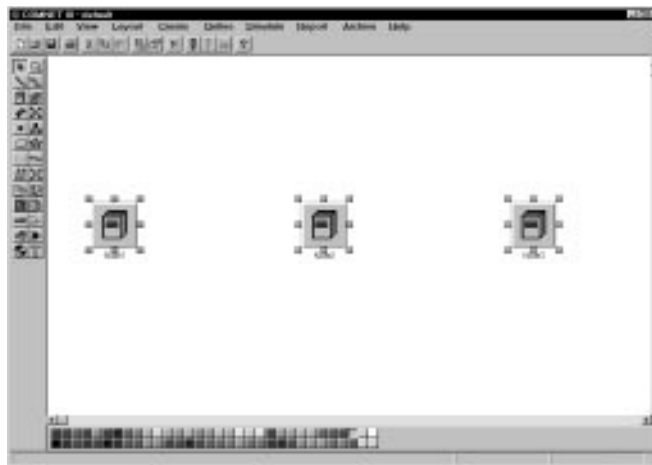
This will distribute three or more selected objects so that they are equally spaced, either in a vertical direction, a horizontal direction, or about a circle.

The screen shots below show the results of using **Distribute** on some unequally spaced objects.

Before using **Distribute**:



After using **Distribute**:

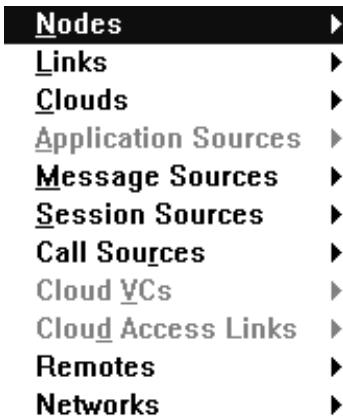




<b>Group</b>	Combines multiple objects into a <i>group</i> . Once objects are grouped, all normal editing operations that can be done on an individual object can be done on the group. This can be useful when your model is divided into logical sections, and you may be happy with the way each section appears, but not with their relation to one another. You can group a section, and then move it with respect to the other sections. You could also cut the group from the layout, or copy and paste it to create an identical group.  If you wish to edit individual components of a group, you will have to use <b>Ungroup</b> .
<b>Ungroup</b>	Breaks up a group into the original component that went into making the group.
<b>Bring to Front</b>	Brings an object that is behind another on the layout to the front so that it can be edited.
<b>Send to Back</b>	Sends an object that is at the front of the model layout to the back, to allow editing of objects that were behind it, or hide behind the other objects.
<b>Foreground Color</b>	Changes the foreground color to the selected color. This will be the color of text labels on objects.
<b>Background Color</b>	Changes the background color of the layout to the selected color. If the color bar is not visible, the new color of the background may be a surprise!
<b>Reset Color</b>	Resets the color of the selected object to the default color.
<b>Snap</b>	If Snap to Grid is turned on, you can only place objects on the layout at grid intersections.
<b>Grid</b>	Places a grid on the model layout as a convenience in aligning objects. The grid has no significance to your model. The grid spacing can be set fo fine, medium, or coarse.

## 5. Creating COMNET III Models

### 5.5.5 Create Menu



The Create menu is used to create new instances of objects to be added to the model layout. These new objects instances are created using either predefined types or types which have previously been defined by the user and placed in a COMNET library. This menu provides a more comprehensive facility for creating model objects than is available from the Tool palette. In particular, it allows the user to create instances of object types that were previously customized by the user and placed in COMNET III libraries using the Archive command on the main menu. Each of the choices in the Create menu pops up an additional menu which contains the list of pre-defined and user-defined objects types.

- |                               |   |
|-------------------------------|---|
| <b>Nodes...</b>               | Presents the user with a list of node types which can be created. Included are pre-defined nodes and those previously defined and placed in the library.  |
| <b>Links...</b>               | Presents the user with a list of link types which can be created. The list includes predefined links as well as those previously defined by the user and placed in the library.   |
| <b>Clouds</b>                 | The cloud provides an abstract model for public or private data networks. The cloud model was designed specifically for frame relay networks, but it may also be used for modeling X.25, TCP/IP, and cell-relay networks. The cloud model is an alternative for modeling WANs instead of explicitly modeling the topology with routers and links. The cloud model consists of the cloud icon and internal detail for access links and virtual circuits. Double-clicking on the cloud icon or using the “enter” menu item in the edit menu will open up the cloud’s internal detail. |
| <b>Application Sources...</b> | Presents the user with a list of application types which can be created. The list includes predefined applications as well as those previously defined by the user and placed in the library.   |
| <b>Message Sources...</b>     | Presents the user with a list of message traffic generator types which can be created. Message sources are used to generate packetized, connectionless message traffic. The list includes predefined generators as well as those previously defined by the user and placed in the library.  |
| <b>Session Sources...</b>     | Presents the user with a list of session traffic generators. Session sources are used to generate packetized, connection oriented data traffic (e.g. Virtual Circuits).   |
| <b>Call Sources...</b>        | Presents the user with a list of call traffic generator types which can be created. Call sources are used to generate circuit switched, connection oriented voice and data traffic. The list includes predefined generators as well as those previously   |

defined by the user and placed in the library.

**Cloud VCs**

A cloud VC represents the virtual circuit or path through the network that connects a source to a destination. In the actual network, the path may consist of a number of intermediate routers and link hops, and the virtual circuit icon models this path in terms of a delay consisting of a propagation delay plus a delay that is the product of the specified number of switches the path contains and the delay per switch. The virtual circuits connect between two access links--there can be at most one virtual circuit between a particular source and destination access link. When a connection is made, the arcs will be a broken to indicate a virtual connection, as opposed to the physical connections used between nodes and links. The arrows on the arcs indicate the direction of information flow. This menu option is only available inside WAN Clouds

**Cloud Access Links**

Cloud access links model the connections between the user and the cloud: this connection may be either the link to DTE or PAD in X.25 or the link to the frame relay access device in frame relay services. Typically, this link will model the local line between the user and the network's point of presence. Important parameters of the access link include the data rate and number of circuits of the link and how much buffer space is available at the network end for traffic arriving at the node. This menu option is only available inside WAN Clouds

**Remotes**

Remotes are used to allow clustering of arcs or sources where the display would be cluttered if all of them were visible. The remote socket also allows a group of sources to share parameters for packetization and assembly delays. It can be used as a message destination.

**Networks**

Used to create transit networks. Transit networks are intermediate networks that interconnect a collection of nodes. They model packets transiting a network.

## 5. Creating COMNET III Models

### 5.5.6 Define Menu

<b>N</b> ode Parameters...
<b>L</b> ink Parameters...
<b>G</b> lobal Commands...
<b>G</b> lobal <b>V</b> ariables...
<b>P</b> rotocols...
<b>A</b> pplication Types...
<b>B</b> ackbone Properties...
<b>R</b> outing <b>C</b> lasses ▶
<b>R</b> outing <b>P</b> enalties ▶
<b>U</b> ser Distributions...
<b>T</b> abular Distributions...
<b>E</b> xternal Traffic ▶
<b>P</b> acket Rate <b>M</b> atrix...

The **Define** menu allows the user to define global model parameters, custom descriptions of processor command repertoires, packet and call routing classes and transport layer lists.

This menu gives COMNET III much of its flexibility by allowing the user to customize most aspects of network operation. The ability to specify many custom classes of network operating parameters allows the user to model disparate network types in one model.

**Node Parameters...** This menu selection brings up a dialog where you can edit the default parameters for the various types of nodes.

**Link Parameters...** This menu selection brings up a dialog where you can edit the default parameters for the various types of links.

**Global Commands...** Presents a dialog box which allows the user to add, remove or edit commands in the model's global command repertoire. These commands fall into 4 categories: Processing, Read, Transport and Write. They are used to model the operation of a computer at a node as it handles traffic. For each command, the user specifies a name and a probability distribution which describes the number of processor cycles each operation consumes. The processing time per cycle is specified for each node in its parameter dialog box.

**Global Variables...** A variable that can be referenced (meaning you can assign to it or query its value) from an application running on any node anywhere in the model. Can save and communicate information during a simulation, allowing you to model state dependencies.

**Protocols...** Presents a dialog box which allows the user to **Add**, **Remove** and **Edit** sets of transport layer protocol parameters, which are then stored with a unique name.

**Application Types...** This feature is used to help define link utilization by application types as seen in the Link report Utilization by Application Types. New application types can be added for message sources, session sources, response sources, and the following commands: transport message, answer message, read file, write file, process data, and setup session.

**Backbone Properties...** Presents a dialog box which allows the user to set the backbone routing protocol for the network.

<b>Routing Classes...</b>	Presents a dialog box which allows the user to <b>Add, Remove</b> and <b>Edit</b> packet or call routing classes which are then stored with a unique name.
<b>Routing Penalties...</b>	Presents a dialog box which allows the user to <b>Add, Remove</b> and <b>Edit</b> packet or call routing penalties which are then stored as penalty tables with a unique name.
<b>User Distributions...</b>	Presents a dialog box which allows the user to define user distributions and give them unique names. A user distribution is simply one of the standard probability distribution functions for which default parameters have been specified.
<b>Tabular Distributions...</b>	Presents a dialog box which allows the user to define table distributions and give them unique names. A tabular distribution allows you to use actual or empirical data to describe the distribution of a random variable. Such data are typically in the form of histograms or frequency plots. The parameters required to build a tabular distribution are described in Appendix B.
<b>External Traffic</b>	Allows you to set up an external traffic file to be used by COMNET III. The external traffic file can be generated by a third party program such as Network General Sniffer or HP NetMetrix.
<b>Packet Rate Matrix</b>	Allows you to directly generate packet traffic without scheduling applications and without producing messages with transport commands. This is useful when data is available on packet rates and sizes, but not much is known about the internals of the application producing the stream of packets.

### 5.5.7 Simulate Menu

<b>V</b> erify Model	
<b>R</b> un Parameters...	
<b>S</b> tart Simulation	<b>F4</b>
<b>H</b> alt Simulation	<b>F5</b>
<b>A</b> nimate...	
<b>T</b> race...	

The Simulate menu gives the user access to simulation controls which specify how the model will be run and which reports will be produced.

<b>Verify Model</b>	Checks the model layout, interconnections and object parameters for consistency.
<b>Run Parameters...</b>	These are the parameters which specify how the model will be run and how long it will run to gather statistics. The parameters are the <b>Replication length</b> , the <b>Warmup time</b> and the <b>Number of replications</b> . In addition, the user can specify whether a Warmup should precede each replication, whether the random number streams should be reset and whether the system should be reset. These parameters and their implications on the results of a simulation run are discussed in detail in the Run Parameters section.
<b>Start Simulation</b>	Starts the simulation. If the model has not yet been verified since the last changes were made, COMNET III will automatically run the verification before starting the model. If the model has not yet been saved since the last changes were made, the user is prompted to save the model.
<b>Halt Simulation</b>	Stops the simulation, if desired, before it executes to completion. Note that this will not occur instantaneously. COMNET III must write all of the requested output reports as part of the Halt process.
<b>Animate...</b>	<p>Allows the user to specify whether the simulation should be animated or not. The user can also specify a simulation time at which the current setting will be toggled. If animation is off, it will turn on and vice versa. This feature can be used to skip “boring” parts while the model is warming up and go to a more interesting part of the simulation.</p> <p>The user can also control the speed of the animation. This is specified with the <b>Step size</b> parameter which can be set in the range between 10 (which is slow) and 1,000 (which is fast).</p> <p>When animation is turned off, COMNET III performs standard discrete event simulation. This means that upon completion of an event, it skips directly to the time of the next scheduled event and performs that event.</p> <p>When animation is turned on, COMNET III adds time steps between each scheduled event so the user will have time to see and appreciate the progress of the animation. This <b>Step size</b> parameter adjusts how long each event is allowed to take during the animation.</p>
<b>Trace...</b>	Provides a step-by-step trace of model execution. The trace can be sent to the screen or to a file in the model directory. The model can be set to stop at each event (single step) or to pause for a specified number of seconds while the trace message is read. When the trace statements are sent to the screen they show in

the status box at the bottom of the screen.

## 5.5.8 Report Menu



### Set File Name...

Allows you to specify a report name for information to be written to. The default name is **Stat1.rpt**. Before any simulation is run you can change the name of the output report so that you can compare the results of one simulation run to another.

This menu lets you turn the various output reports for your model on or off. Individual menu items allow specification of reports for each of the following items:

- Nodes**
- Links**
- WAN Clouds**
- Application Sources**
- Message and Response Sources**
- Call Sources**
- Session Sources**
- Transport Commands**
- Setup Commands**
- Global Transport and Answer Commands**
- Global Setup Commands**
- Background Packet Flows**

To view the reports, choose the following menu item:

### Browse Reports...

This item presents a dialog box which allows the user to specify which replication of reports to display to the user. When the user clicks on VIEW the system editor of the operating system is invoked to display the requested reports

### Select Snapshots...

This feature allows you to display utilization percentages for nodes, links, and cloud virtual circuits on the model during a simulation run. Threshold limits can be set along with color identification for alarm conditions that will turn a particular node, link or virtual circuit the color of the alarm condition. Utilization percentages are updated every 5 seconds during a simulation.

### Take Snapshot...

This feature allows you to take a snapshot of a particular node, link, or virtual circuit at any time during the simulation. In order for this feature to work, you must first turn on the snapshot report for the particular node, link or virtual circuit you are interested in. Then during the simulation, you need to highlight the object and from the Report Menu select Take Snapshot. The simulation will be halted and the information displayed to the screen.

For more information about the reports, see Chapter 8. Reports.



### 5.5.9 Library Menu



#### **Edit Contents...**

Allows you to define custom classes of COMNET III objects and manage libraries of these objects. New object types can be created by copying existing objects, elaborating the attributes of the object to define a new type then saving this new object type to a library. The objects shown in this list are available to all COMNET III models; they are not limited to the current model.

#### **Software Objects**

Allows you to load or unload the advanced features pack of COMNET III. You will only be able to load the advanced features pack if you have purchased it when you purchased COMNET III.

## 5. Creating COMNET III Models

### 5.5.10 Help Menu



The Help menu provides access to the COMNET III help system. In the present version this capability is available only on the Windows version.

<b>Index...</b>	Provides access to Windows online help.
<b>Dump Memory Usage...</b>	Shows the current amount of memory that COMNET III is using.
<b>About COMNET III...</b>	Displays the current release number and build number for COMNET III. ReadMe File...
<b>ReadMe File...</b>	Displays a readme file with the latest release information and notes about COMNET III.
<b>Reference Manual...</b>	Launches the online Reference manual which is found on the installation CD ROM. In order to access the online Reference manual, the COMNET III CD ROM must be in the CD ROM drive.

## 5.6 Program Operation Aids

### 5.6.1 Tool Tips and Status Bar Messages

When you position the mouse cursor over an icon on the palette or toolbar, a short message pops up after a brief delay to indicate the icon function. Messages are also provided in the status bar after you click on a palette or toolbar icon or when you scroll through menubar items. The UNIX versions provide status bar messages, but not the pop-up tool tips.

The palettes and toolbars can be located anywhere on your screen. The color palette, menu toolbar, and 3D toolbar can be toggled on and off.

### 5.6.2 Multiple Report Windows

The Report/Browse Reports menu item now allows you to open multiple asynchronous windows for browsing reports. It is no longer necessary to close the report file editor before continuing with other COMNET operations.

### 5.6.3 3D View

A 3D view showing all levels of the network hierarchy can be displayed in a separate window. There is a separate toolbar of controls for moving the camera location so you can view the network from any perspective. Trace highlighting and alarms are displayed in the 3D window if the window is open during the simulation. At the present time, the 3D view is available only on the Windows NT version. Support for 3D graphics is expected soon for Windows 95, at which time the 3D view capability will also be available on the Windows 95 version of COMNET III.

### 5.6.4 Dockable Palettes and Toolbar

The palettes and toolbars can be located anywhere on your screen. The color palette, menu toolbar, and 3D toolbar can be toggled on and off.

### 5.6.5 Arc Editing

The joints on an arc can be added, moved, or deleted without cutting and redrawing the entire arc. Arc joints are now rounded to distinguish between a corner belonging to a single arc and a corner created by the intersection of 2 separate arcs. Users can now set arc color using the color palette.

### 5.6.6 Color Palette

The color palette can be used to paint icons, arcs, text, or various miscellaneous shapes. It is also used to set the foreground and background colors.

### 5.6.7 Shapes

A variety of shapes can be added to the display and edited, combined, and colored to form your own icons. These icons can be used as background art for your network layout or can be grouped with model objects to customize the appearance of objects.

### 5.6.8 Bitmap Import

Using File/Import/Bitmap, you can now directly import your own bitmap icons for nodes, links, background maps, and other objects. Your icon is automatically added to the list of icon choices for the designated object class. It is no longer necessary to use SIMDRAW to import bitmap icons.

### 5.6.9 Export Encapsulated PostScript

The File/Export/Encapsulated PostScript menu item produces an image of the entire network layout. On UNIX versions of COMNET, this item is the same as File/Print. On Windows versions, File/Print produces a native Windows print of the layout visible on the screen.

## 5. Creating COMNET III Models

### 5.6.10 Export Raster Bitmap

File/Export/Raster Bitmap dumps the screen image to a bitmap file.

### 5.6.11 Background Text

Users can now set the color, size, and font of background text.

### 5.6.12 Model Editing Features

#### View by List:

The View by List option of the Edit menu allows you to create a list of all model elements or a subset of model elements. By navigating the list, you can find an edit a particular element, or attributes of the element. The View by List is handy for navigating a model that is either very large or has numerous subnetworks or both.

By selecting the **Filter Items with Default Value** all attributes whose values have not changed from their default setting are filtered out of the list. This is convenient if you are interested in viewing only the user defined data that you have entered.

By selecting the **Filter "DEFAULT" Objects**, all objects with names like DEFAULT and STATNDARD, i.e. those provided in the default parameter sets will be filtered out.

#### Select by List:

The Select by List option of the Edit menu allows you to select model elements using an interface very similar to that of View by List. The Select by List is handy for navigating a model that is either very large or has numerous subnetworks or both.

The Select by List allows multiple selection, but all of the selections must be at the same hierarchial level, i.e. in the same subnetworks.

#### Find:

The Edit menu now has a Find operation, which is also easily invoked using CTRL+F. Find is handy for locating a particular model element in a model that is either very large or has numerous subnetworks or both.

Using **Matching Rules** you can search for an element by name, you can search for an element having a particular parameter set, or you can search for an element having a particular attribute.

Using **Options** you can elect to match case, to match all strings containing the search string, or to match strings containing a pattern. The pattern must be of the form [part1]\*[part2]. Both parts are optional. [Part1]\* matches any string beginning with part1, \*[part2] matches any string ending in part2, and [part1]\*[part2] matches any string that begins with part1 and ends in part2.

If you check the "Search hidden items too" and if the Find operation matches a model element that's been hidden via the Hide option of View menu, the found item will be unhidden.

In all cases the result of the Find selects the found element, and if the element is located in a subnetwork that is not the currently open subnetwork, the editor will move to the appropriate subnetwork.

### 5.6.13 Report Request Manager

The report request manager provides an interface to selectively turn on or off certain reports for individual items in the model or for the entire model. The report request manager presents the possible reports in a hierarchical list arranged by type of report. The reports may be selectively added to specific model elements by editing the reports or the reports may be turned on or off for everything with a single button in the dialog.

## 6. Advanced Software Modeling Features

### 6.1 Introduction

COMNET III release 1.3 has added a number of advanced features for modeling distributed software systems. After installation, the Advanced Software Modeling Features will be disabled. The reason for this is to simplify model formulation for novice users and to steer users away from the more complex aspects of COMNET III unless such capabilities are essential for the performance measures of interest.

**To Access the COMNET III Advanced Software Modeling Capabilities:**

- 1) **Select the *Library* menu option from the Main Menu Bar**
- 2) **Select the *Software Objects* option**
- 3) **Select the *Load* option**

The Advanced Software Modeling features are now available for all models. If you had existing models designed with an earlier release of COMNET III, and those models used application sources you would need to load the Advanced Software Modeling features before the previous models would load.

The Advanced Software Modeling capabilities will be described in detail in the following sections.

### 6.2 User Variables

User variables can save and communicate information during a simulation, allowing you to model state dependencies.

#### **Variable Scope:**

Scope” is a term borrowed from the world of programming languages, and it means when and where a variable can be referenced. A variable is known in a scope in much the same way as an internet address is known in a domain.. User variables have one of three scopes: global, node, and application.

A global variable can be referenced (meaning you can assign to it or query its value) from an application running on any node anywhere in the model.

A node variable can be referenced only on the node on which it was defined.

An application variable is a very special kind of variable. Each application has one implicitly, and only commands of that application can reference it. If you are conversant in programming terminology, an alternative way to understand application variables is to imagine that a command in a command sequence can generate an output parameter, and that output parameter can flow downward to subsequent commands’ input parameters. The Assign Variable command has an implicit output parameter, and every command has an implicit input parameter.

If you have a global variable and a node variable with the same name, and an application references that name, the node variable will be referenced. When matching names to variables, node variables are always checked first.

#### **Defining Variables:**

A global variable is defined by selecting the User Variables option from the Define menu. A node variable is defined by clicking the Variables button on the node properties dialog. Application variables are implicitly present on appli-

## 6. Advanced Software Modeling Features

cations, and need not be defined.

When defining a variable, you give it a name, and you specify its type. The type dictates what data that can be stored in the variable, and will be one of the following: integer, real, boolean, and string. Integer and real variables hold integer and real numbers, respectively. Boolean variables hold the value True or False. String variables hold text, much like the message text of a transport command.

### Initial Values:

At the beginning of simulation, all variables are initialized with default values. Integer and real variables will be set to zero, boolean variables will be set to false, and string variables will be made empty.

### Assigning Values:

During the simulation, a value is given to a variable using the Assign Variable command.

### Querying Values:

During the simulation, you can query the value of a variable in a number of different contexts. For example, you can use the value of a variable as part of a message size calculation. To do so, simply use the variable's name in an expression (see the section on Expressions).

### Variable Errors:

During the simulation, any invalid operation on a variable will be reported as a variable error. A variable error is similar to a disk access error, and is reported similarly. Unless you have turned it off, the Variable Error snapshot report will be active, and its alarm will be set to sound whenever errors are greater than zero.

### Exclusive Access and Semaphore Behavior:

Because applications in your model can execute simultaneously, it's very conceivable that more than one application might reference the same global or node variable simultaneously. Depending on the logic of the model, you might be concerned about one application assigning to the variable while another is querying it, or two applications assigning to the variable at the same time.

To prevent simultaneous access of a variable, set the Exclusive Access checkbox, and give an access time. The exact value of the access time is probably incidental to the simulation, the important thing is that only one application will be allowed to access the variable (to either assign or query) during that access time. The access time serves to induce ordering on events that would otherwise occur simultaneously.

Using the exclusive access option, it's possible to create logic whereby several applications are waiting for a variable to achieve a certain value, but when it does, only one of the waiting applications executes, and it immediately changes the variable's value, thus ensuring that the other applications will continue to wait. This simultaneous waiting behavior is common in semaphore modeling.

Just like in the real world, it's possible for applications to vie for overlapping sets of variables and deadlock. In COMNET III, such deadlocks are temporary, because each side will try only a finite number of times, as specified in the "max no. of tries" field of the variable. After that number of tries, a variable error will be generated, and the application will continue on as if it did obtain access.

### Troubleshooting:

Using variables to control model behavior can lead to complex interactions, and offers tremendous potential for mis-

takes in the model. To help find unintended variable behavior, turn on the Variable Value snapshot. You must specify one variable to track, then the snapshot will show you the value of the variable as it changes. The value shown on any node on which the snapshot is enable will correspond to the value of the lowest scoped variable matching the name you specified. For example, suppose you have a global variable named FOO, and on Node1 you also have a node variable named FOO, and you enable the Variable Value snapshot for Node1 and Node2. The snapshot on Node1 will reflect the value of the node FOO, while the snapshot on Node2 will reflect the value of the global FOO.

Also handy for troubleshooting is the old Trace To File option, available via the Trace option of the Simulate menu. Every variable assignment causes an informative trace message to be generated.

## 6.3 New Commands

### Assign Variable Command:

Use the Assign Variable command to give a value to a variable. The variable combo box on the command properties dialog will allow you to specify the target of the assignment. Select a global or node variable by name, or specify the application variable by selecting “output parameter.”

The “two-dot” button next to the variable combo box allows you to add variables. If you’re editing a node command, you’ll be permitted to add node variables, and if you’re editing a global command, you’ll be permitted to add global variables.

Specify the value for the assignment in the text box. The value can be any expression, involving constants, other variables, random number distributions, and intrinsic functions.

### Wait For Command:

The Wait For command is a generalization of the old Filter Messages command. In fact, by default, the Wait For command functions exactly the way Filter Messages did. Wait For evaluates a condition, and if the condition is met, it completes and gets out of the way for the next command to run. If the condition is not met, Wait For blocks until the condition is met, or until the time-out timer expires, whichever comes first. By “block,” we mean the application running the command sequence is suspended.

The condition can be of three types: Received Message, Alarm, or Expression.

#### *Wait For Received Message:*

Behaves exactly as the old Filter Messages command.

#### **Wait For Alarm:**

If the specified alarm is already sounding when this command executes, no blocking occurs, the command completes and the next command is executed. If the alarm is not already sounding, the command blocks until the alarm does sound.

The alarm combo box shows all the alarms that are currently set. Use the two-dot button to set additional alarms if necessary, and pick an alarm. Then specify the node or link on which the alarm must sound.

## 6. Advanced Software Modeling Features

### *Wait For Expression:*

The expression can be almost any expression, involving constants, user variables, and most of the intrinsic functions. Some intrinsic functions, like SimTime, are prohibited. The expression must evaluate to a boolean value.

When the command is executed, the expression is evaluated, and if it is TRUE, no blocking occurs, the command completes and the next command is executed. If the expression is FALSE, the command blocks until the expression becomes TRUE. Of the limited number of things permitted in a Wait For expression, only user variables are liable to change after the expression is evaluated the first time. Thus, the expression is only re-evaluated if and when referenced variables change.

Clearly, a Wait For expression that does not involve one or more variables is not of much use, UNLESS the “stop waiting” option is checked. If the “stop waiting” option is checked, then either the expression is FALSE when first evaluated, or the command will block for the duration of the time-out time.

### *Stop Waiting:*

The blocking process you’re modeling will probably not wait forever. To model a process that times out, check the “stop waiting after” option, and give the time limit you’re willing to wait. After that time limit, the command and the application will be terminated. To catch this event and do something special in response to it, click the Triggered Sources button on the application source, select the source that will perform the special response, and use the “upon wait timeout” triggering rule.

### *Troubleshooting:*

With the increased flexibility of the Wait For command comes increased complexity and potential for mistakes in the model. To assist in finding wayward applications that are in perpetual Waits, turn on the Suspended/Pending Apps snapshot. This snapshot will show you a count of suspended and pending applications on any node on which it’s enabled.

### **Macro Command:**

Programmers can think of the Macro command as a subroutine. The Macro command is a handy way to organize commands that are frequently used together.

### *Command Sequence*

A Macro command contains a command sequence, similar to that of an application source. Other commands from the command repertoire, or global commands, can appear in this command sequence. When a macro command is executed, each of the commands in its command sequence is executed.

### *Looping*

A Macro command can be used to implement a simple looping behavior in an application source’s command sequence. If a Macro command has a “number of executions” value greater than 1, the Macro command will execute multiple times, and it will be as if the commands in it were being looped.

Hint: To set the loop count on the fly, you can use “input parameter” for the number of executions of the Macro command, and add an Assign command higher up that computes the value.



***Atomic Execution***

Not a high-tech form of capital punishment, but nevertheless a feature for advanced users. You might need this if you're modeling a semaphore-like behavior, for example you have multiple applications waiting for a variable to become zero, and you want only one of them to resume execution when the variable does become zero. To accomplish this, you need to ensure that only one application accesses the variable at a time. No problem, you enable the "exclusive access" option on the variable. But you also need one application to 1) query the variable and find that it's now zero, and 2) change the variable to something non-zero. Actions 1 and 2 translate into a Wait For command and an Assign Variable command, and somehow you need to do both of them without another application getting in between. For example, after the first command is done seeing that the variable has become zero, another application could get a chance to query the variable before you have a chance to change it.

For this example, what you need is a way to say "do commands 1 and 2, and don't let anyone else touch the variable until they're both done." To keep everyone else away from the variable for the duration of the two commands, put both commands in a Macro command, and check the "atomic wrt variables" checkbox on the Macro command.

There can be more than one variable referenced in the commands of an atomic Macro's sequence. As each variable is accessed, it becomes locked for use only by this application, and remains locked until the Macro completes. Use the atomic feature with care, it's very easy for two Macro commands that need overlapping sets of variables to execute simultaneously and deadlock (just like in the real world!). Note that, in COMNET III, deadlocks are only temporary, because after the number of tries specified on the variable, one party will give up.

**6.4 Expressions**

Expressions are used to give a value to a variable in an Assign Variable command, to specify a condition for which to wait in a Wait For command, and to specify a random number distribution in a User Distribution. You type them free-form into a text box. The verification that takes place for the dialog will verify that you have no syntax errors, but it will not tell you that you misspelled a variable name, or that you're trying to add 1 to a string variable. The latter sorts of errors will be detected when you Verify Model.

Expressions are either numeric, string, or boolean.

**Numeric Expressions**

Formally speaking, a numeric expression is an algebraic expression made up of one or more terms combined using + and - operators, and each term is made up one or more factors combined using \* and / operators. A factor boils down to either a pair of parentheses containing a numeric expression, or one of the following: a constant, a random number distribution, a variable, or an intrinsic function returning a numeric type.

***Constants:***

Integer constants look like this: 1, 2, 3. Real constants require a decimal point, like this: 1.0, 2., 3.5.

***Random Number Distributions:***

These are the things you're used to seeing in combo boxes on property dialogs, for specifying inter-arrival times, message sizes, etc. You enter them in an expression exactly as you would in a combo box. The only difference is that you have to remember the format, there is no combo box to help you (except for the expression field of a User Distribution, where you CAN get some help from a combo box).

Note that User Distributions can be used here as well.

At the moment the expression is evaluated, a random number is drawn from the sample. Random Number Distribu-

## 6. Advanced Software Modeling Features

tions are not permitted in a Wait For expression.

### *String Expressions:*

String expressions are commonly used to assign a value to a string variable, or to generate a string to be used as message text. A string expression is made up of one or more strings concatenated together via the + operator. Strings are either literal text between double quotes, string variables, or intrinsic functions that return String type. An empty string is denoted by "".

### *Boolean Expressions:*

Boolean expressions are commonly used to assign a value to a boolean variable, or as the condition of a Wait For command. A boolean expression is made up of one or more terms combined using the OR operator, and a term is made up of one or more factors combined using the AND operator. A factor boils down to either a pair of parentheses containing a boolean expression, a boolean variable, a boolean constant, or an intrinsic function returning Boolean type. A boolean constant is one of TRUE or FALSE, and must be all uppercase.

### *Intrinsic Functions:*

Some take arguments, and all return handy values, similar to the way a random number distribution returns a value. Some are not permitted in a Wait For expression.

<b>Name</b>	<b>Arguments</b>	<b>Return type</b>	<b>OK in Wait For</b>
ABS	Integer or Real	Integer or Real	yes
FLOAT	Integer	Real	yes
INTTOSTR	Integer	String	yes
LOWER	String	String	yes
MAXOF	Integers or Reals	Integer or Real	yes
MINOF	Integers or Reals	Integer or Real	yes
ODD	Integer	Boolean	yes
POSITION	String	Integer	yes
REALTOSTR	Real	String	yes
ROUND	Real	Integer	yes
STRLEN	String	Integer	yes
STRTOINT	String	Integer	yes
STRTOREAL	String	Real	yes
SUBSTR	(1)	String	yes
TRUNC	Real	Integer	yes
UPPER	String	String	yes
SimTime	none	Real	no
SELECT	(2)	any	yes
InputParameter	none	Real or String	yes
OriginalMsgSize	none	Real	yes
OriginalMsgText	none	String	yes
OriginalFileSize	none	Real	yes

(1) SUBSTR (pos1, pos2, string), pos1 and pos2 are integer expressions, returns portion of string between pos1 and pos2 inclusive.

(2) Select (bool, exp1, exp2), bool is a boolean expression, exp1 and exp2 are expressions of any type. If bool is TRUE, returns value of exp1, otherwise returns value of exp2.

**Examples:**

“Message to “ + StrVar1

ABS(Exp(10.0) - 2.0)

MsgCountVar \* MsgSizeVar

(MsgCountVar > 10) AND (SesCountVar > 10)

### Generalized User Distributions

User Distributions have been generalized in 1.3. In 1.2, they were customized random number distributions, in which the user had set the parameters of the distribution. There was no way to shift or clip the distribution, your shape manipulations were limited to those made by changing the parameters.

In 1.3, you can still create User Distributions that have only parameter changes, or you can also give an arbitrary expression, and of course this expression can feature a random number distribution. So, for example, you can clip an exponential distribution so that it returns only values less than 12:

MAXOF(Exp(10.0), 12)

Note that the expression may also involve user variables, and it need not involve any random number distributions, in which case a sample from the distribution returns a state dependent number rather than a random number.

Hint: If you’re planning to create a shaped distribution based on a random number distribution, first select the “Sample only option” and pick the random number distribution using the combo box. Then switch to the “Expression” option. When you switch, the text of the sample combo will be copied to the expression text box, giving you a head start.

### Variable Execution Count for Commands

While specifying the command sequence of an application source, the number of executions for each command may now be a Sample. The sample may be one of the standard random number distributions. The sample may be a User Distribution, in which case it might be a state-dependent number instead of a random number. The sample may be “input parameter,” which means a number generated by an Assign Variable command occurring earlier in the command sequence.

## 6. Advanced Software Modeling Features

# 7. Automatic Parameter Iterator

## 7.1 Introduction

Before the release of COMNET III version 1.3, model experimentation was conducted in the following way:

- 1) Manually change something in the model.
- 2) Manually change the name of the report file and associated statistics files using the Set File Name option of the Reports menu.
- 3) Run the simulation.
- 4) Go to step 1, and repeat for all the intended changes.
- 5) Manually compare the statistics collected for the various runs.

In COMNET III release 1.3, the above steps can now be automated.

## 7.2 Terminology

### *Experiment:*

Steps 1-5 (listed above) are repeated for every change, and the resulting sets of output files.

### *Scenario:*

Steps 1-5 (listed above) run once, correspond to only one change in the model.

### *Factor:*

The thing in the model being changed.

### *Level:*

One of the values given to the factor in step 1.

## 7.3 Specifying an Experiment

Select the Run Parameters option of the Simulation menu. There is a combo box labeled “experiment on...” and initially it indicates “none,” or no experiment specified. The other option available in the combo box is “Select factor...” This combo box functions something like a pop-up menu, because when you select “Select factor...” you are taken to another dialog, where you select a factor.

### **Select Factor:**

The factor selection dialog is very similar in operation to the View By List dialog, and as a matter of fact you can use View By List as an alternative method of specifying an experiment.

## 7. Automatic Parameter Iterator

Find the attribute on which you want to experiment. You will notice right away that relatively few attributes are available for experimentation. We've tried to provide experimentation on the attributes that most directly influence traffic level and delay.

### Levels:

After you've selected the attribute, click the Factor button. You will be asked to give the experiment a name, and to specify levels. The current value of the attribute is offered as the default level. Change it and/or add more values, separated by semicolons. When you click OK, the validity of each level will be tested, and any obvious errors will be reported. Note that it's possible to give unsuitable values via this interface, because the normal checks performed during normal dialog interaction are not performed.

### Apply to All:

An option available for some factors is the "Apply to all" option. If you check it, then the levels you give will be applied to all model entities of the same class. For example, if the factor you specify is Bytes or Cycles of some message source, you have the option of applying the levels only to that message source, or to all the message sources in the model.

### Changing the Experiment:

When you're satisfied that you've picked the right factor and given the right levels, click Done. The combo box on the Run Parameters dialog will then indicate the currently selected factor. To clear the experiment, select "none" again. To change it, select "Select new factor and/or levels"

The experiment you specified will be saved with the model.

## 7.4 Running an Experiment

To run the experiment, simply select Start Simulation or click the green light button, as you normally would. The iterator automatically assigns your first level, runs the simulation, and so on. After the run, the iterator will restore the value of the factor to the value it had before the experiment.

## 7.5 Analyzing the Results of an Experiment

Statistics and reports files will be named automatically; for example, Expt1-1.rpt (substitute your experiment name for Expt1). You can browse reports as is normal, but you'll have more report files than normal to look at.

### Familiar Territory:

To examine statistics, go to the properties dialog of the model element whose statistics you want to examine, and click the Statistics button, as usual. You will be asked the question, "Statistics from which experiment?" and Expt1 (or your experiment name or names) will be in the list. After selecting the experiment and clicking Continue, you will go to the Statistics Requests dialog, where you can select a statistic and click View to see the data and get plots, as usual.

If you use multiple replications in your model, you probably need some orientation, as there are now  $N \times M$  statistics, where  $N$  is the number of scenarios (= number of levels) in the experiment, and  $M$  is the number of replications in each scenario. The stats table will display a row for each replication, showing min, mean, and standard deviation for the statistic during that replication. You will see  $M$  rows of data, but all the data applies only to the first scenario, Expt1-1. To switch to the data for the second scenario, select Expt1-2 from the "Stat file names" combo box.

To plot the statistic for one replication from one scenario, make sure the right scenario is showing, select the row of the desired replication, and click the Plot button.

### **All Checkbox:**

For the most part, the interactions above are familiar territory, they're not much different from what most users experienced in release 1.2. The All checkbox is not new in 1.3, but its utility in combination with a multi-scenario experiment is great.

When you check the All checkbox, the per-replication data goes away, and the data that replaces it consists of one row for each scenario, showing the summary min, max, and std. dev. for that scenario, averaged over all the replications in that scenario. In the All table, you see a side-by-side comparison of each scenario, so you can quickly appraise the effect of the different levels on this statistic.

Now when you click the Plot button, you are given a bar graph showing how the mean of the statistic varies by scenario.

## 7. Automatic Parameter Iterator



## 8. Reports

The following reports are produced by COMNET III:

### ***Node Reports***

- Processor + Disk Utilization
- Input Buffer Use by Port
- Input Buffer Totals
- Output Buffer Use by Port
- Output Buffer Totals
- Received Message Counts
- Disk Access Error Counts
- Session Level
- Call Counts
- Call Level

### ***Link Reports***

- Channel Utilization
- Utilization by Application
- Utilization by Protocol
- Collision Statistics
- Session Level
- Call Counts
- Call Level

### ***WAN Cloud Reports***

- Frame Delay by Virtual Circuit
- Frame Counts by Virtual Circuit
- Access Link Statistics

### ***Application Source Reports***

- Application Run Length

### ***Message + Response Source Reports***

- Message Delay
- Packet Delay

### ***Call Source Reports***

- Blocked Call Counts
- Disconnected Call Counts

## 8. Reports

- Preempted Call Counts

### ***Session Source Reports***

- Message Delay
- Packet Delay
- Setup Delay
- Session Length
- Setup Counts

### ***Transport + Answer Commands Reports***

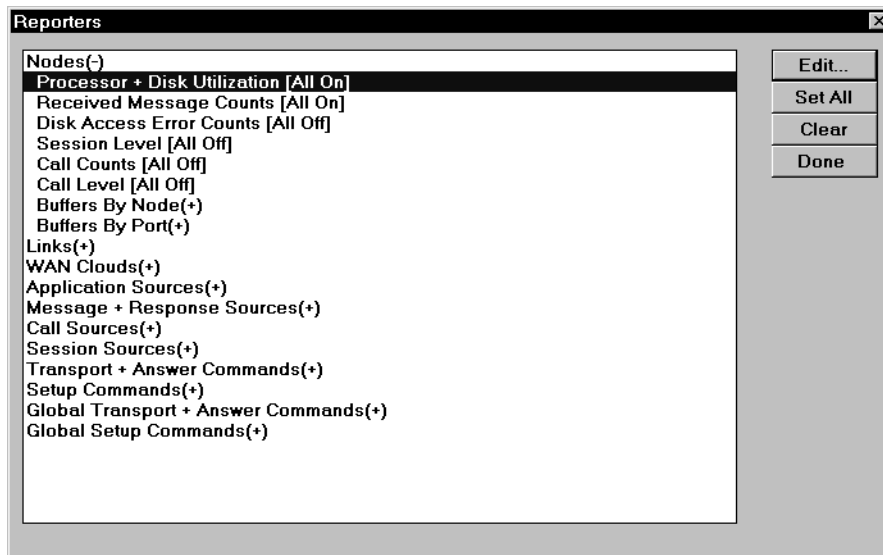
- Message Delay
- Packet Delay

### ***Setup Command Reports***

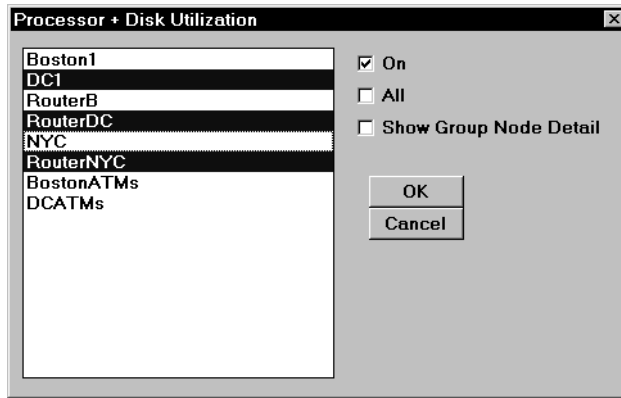
- Message Delay
- Packet Delay
- Setup Delay
- Session Length
- Setup Commands

These reports are produced at the end of each replication of the model. They can be selectively turned on by choosing the menu item Report and then selecting the appropriate report from the pulldown and flyout menus. A dialog box will appear where reports can be turned on or off for each individual model element (node, link, etc.). Here is an example of one of these dialog boxes.

Statistical monitoring during the simulation is active only for the statistics that need to be included in formatted reports (or viewed through statistics buttons). By requesting fewer reports and including fewer objects in reports, you can speed up the simulation.



Reporters dialog box



### Selecting Reports on Processor and Disk Utilization

These reports are prepared as soon as the simulation run completes. They are in standard text format and are formatted to fit within 80 columns for easy viewing and printing. The Report/Browse Reports menu item may be used to view the reports once the simulation is complete, or any text editor may be used. The reports are placed in a subdirectory which has the same name as the model. Within the subdirectory the all report replications are in the file Stat1.rpt,. If you run a simulation again this file will be overwritten with the new reports for the next run.

## 8. Reports

### 8.1 Node Utilization

This report presents a summary of the utilization rates for the processors on each node. Data for computer group nodes is aggregated. If individual node statistics are desired for the computer group nodes, simply set the “Show Group Node Detail” item when setting the report.

CACI COMNET III

Thu Feb 24 11:42:27

PAGE 15

NODE UTILIZATION  
REPLICATION NUMBER 1

NODE	DISK	DISK USAGE (KILOBYTES)			PROCESSOR
	REQSTS GRNTED	AVERAGE	MAXIMUM	STD DEV	% UTIL
Server1	0	0.000	0.000	0.000	0.00
Server2	0	0.000	0.000	0.000	0.00
StationATM	0	0.000	0.000	0.000	0.00
Station1	0	0.000	0.000	0.000	0.00
Station2	6	119.617	130.000	10.375	0.06
Server3	0	0.000	0.000	0.000	0.00
Station3	0	0.000	0.000	0.000	0.00
Server4	0	0.000	0.000	0.000	0.00
Station4	0	0.000	0.000	0.000	0.00

Node	The name of the node.
Disk Requests Granted	Disk requests are issued by read and write commands to access files stored on the local disk of a C&C or Computer Group node. The number of accesses made is reported. An interrupted read or write command (because of time slicing or node failure) will count as multiple access.
Average Disk Usage (KB)	Files are stored on the disk. This is the time weighted sum total of the size of all the individual files.
Maximum Disk Usage (KB)	The peak sum total of the size of all the individual files.
Standard Deviation Of Disk Usage (KB)	The standard deviation of disk usage about the average. Quoting a standard deviation should not be interpreted that the measured value is normally distributed about the average—it probably is not.
Processor Utilization	At any instant in time the node processor is either idle or busy. The processor is counted as busy when executing process commands, read and write commands are accessing files, or packets are being created by message, session or response generators. When a packet is created the processor is made busy for the packetizing delay, followed by the packet switching time. In addition, the processor is busy for the switching time when packets are routed across the node.

## 8.2 Application Delays

This report presents a summary of delays experienced by applications executing on nodes. For each node, the list of applications running on that node is summarized. Data for computer group nodes is aggregated. If individual node statistics are desired for the computer group nodes, simply set the “Show Group Node Detail” item when setting the report.

CACI COMNET III

Thu Feb 24 11:42:27

PAGE 14

APPLICATION DELAYS  
REPLICATION NUMBER 1

NODE: APPLICATION LIST	NUMBER COMPL	APPLICATION DELAY (MILLISECONDS)			
		AVERAGE	MINIMUM	MAXIMUM	STD DEV
Station2: GeneralApp	3	7.88	7.48	8.28	0.33
Station3: Poll Reply	2	0.00	0.00	0.00	0.00
Station4: Reply App	1	0.00	0.00	0.00	0.00

Node	The name of the node. The applications are listed by their host node.
Application List	The names of the applications that run on the node.
Number Complete	The number of instances of the application that have completed execution. This means that they have completed the last command in their command list. It is possible that, at the instant the report was produced, applications were still scheduled for the node. These applications will not be included on the report as they have not completed.
Average Application Delay (Milliseconds)	The average time it took to execute the completed applications, counted from the point at which they were first scheduled until the point at which their last command completes. This includes time spent waiting in the application pending list of the node while other applications are executing.
Minimum Application Delay (Milliseconds)	The minimum of the application delay for completed instances of the application.
Maximum Application Delay (Milliseconds)	The maximum of the application delay for completed instances of the application.
Standard Deviation Of Application Delay (Milliseconds)	The standard deviation of the application delay for completed instances of the application. Quoting a standard deviation should not be interpreted that the measured value is normally distributed about the average—it probably is not.

## 8. Reports

### 8.3 Received Message Count

This report presents a count of received messages for each destination node listed by message name. Data for computer group nodes is aggregated. If individual node statistics are desired for the computer group nodes, simply set the "Show Group Node Detail" item when setting the report.

CACI COMNET III

Thu Feb 24 11:42:27

PAGE 9

RECEIVED MESSAGE COUNTS  
REPLICATION NUMBER 1

RECEIVER	COUNT	MESSAGE NAME
StationATM	9	MsgATM
StationATM	6	SessATM
Station1	9	MsgATM
Station1	6	SessATM
Station2	1	Aloha Session
Station2	2	Poller Message
Server3	1	Poller Message
Station3	2	Poller Message
Server4	2	Aloha Session
Station4	1	Aloha Session

Node                                   The name of the receiving node.

Count                                   The number of messages received during the simulation.

Message Name                           The name of the received message. This is not the message text but the name of the message, session, answer, or transport command that sent the message.

## 8.4 File Warnings

When there are problems with file I/O at any of the processing nodes, COMNET does not stop the simulation. Instead, it keeps a count of these problems and presents them in this report. Data for computer group nodes is aggregated. If individual node statistics are desired for the computer group nodes, simply set the “Show Group Node Detail” item when setting the report.

CACI COMNET III

Thu Feb 24 11:42:31

PAGE 29

### FILE WARNINGS REPLICATION NUMBER 1

NODE NAME	# TIMES STORAGE CAPACITY EXCEEDED	# TIMES NO READ FILE EXISTED	# TIMES READ BYTES IN FILE EXCEEDED
Server1	0	0	0
Server2	0	0	0
StationATM	0	0	0
Station1	0	0	0
Station2	0	0	0
Server3	0	0	0
Station3	0	0	0
Server4	0	0	0
Station4	0	0	0

**Node** The name of the node which is reporting the error.

#### Number Of Times Storage Capacity Exceeded

When write commands are executed on a C&C node or a Computer Group Node it is possible that the local disk capacity modeled for the node will be exceeded. The host node and the disk are counted as busy for the length of time it takes to execute the requested transfer and the number of times this occurred reported. At the end of the transfer the disk will be full but its size will not be increased.

#### Number Of Times No Read File Existed

When read commands are executed on C&C or Computer Group nodes it is possible that the requested file does not exist at that instant in time. The host node and the disk will be counted as busy as if the transfer took place, but the file will still not exist at the end of the transfer. The number of times this occurred is reported.

#### Number Of Times Read Bytes In file Exceeded

When read commands are executed on C&C or Computer Group nodes it is possible that the requested file is not as big as the requested transfer. The host node and the disk will be counted as busy as if the full transfer took place, but the file will not be incremented in size at the end of the transfer. The number of times this occurred is reported.

## 8. Reports

### 8.5 Node Buffer Policy

A report is available on buffer policy actions either on a port-by-port basis or aggregated over the entire node.

CACI COMNET III RELEASE 1.2 BUILD 212 Thu Mar 07 12:54:04 PAGE 5

Acme2

NODES: INPUT NODE BUFFER POLICY

REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

NODE	PREEMPTION		THRESHOLD	
	PACKETS PREEMPTING	PACKETS PREEMPTED	PACKETS EXCEEDED	PACKETS REJECTED
Boston1	0	0	1	0
DC1	0	0	0	0
RouterB	0	0	0	0
RouterDC	0	0	0	0
NYC	0	0	0	0
RouterNYC	0	0	0	0
BostonATMs	0	0	0	0
DCATMs	0	0	0	0



## 8.6 Link Delays & Utilization

This report provides utilization rates for links used to carry connectionless and virtual circuit messages.

CACI COMNET III

Thu Feb 24 11:42:30

PAGE 27

### LINK DELAYS AND UTILIZATION REPLICATION NUMBER 1

LINK	FRAMES		TRANSMISSION DELAY (MS)			% UTIL
	DELIVERED	RESENT	AVERAGE	STD DEV	MAXIMUM	
<b>Link3</b>						
FROM Server1	0	0	0.000	0.000	0.000	0.00
FROM Router1	0	0	0.000	0.000	0.000	0.00
<b>Link2</b>						
FROM Router1	6	0	418.750	414.583	833.333	4.19
FROM Server2	9	0	556.944	390.873	833.333	8.35
CSMA/CD	42	0	0.601	0.348	0.821	0.04
Ring	15	0	0.301	0.244	0.500	0.01
<b>LinkB</b>						
FROM StationATM	246	0	0.333	0.000	0.333	0.14
FROM ATM	246	0	0.333	0.000	0.333	0.14
<b>LinkA</b>						
FROM Server1	246	0	0.333	0.000	0.333	0.14
FROM ATM	246	0	0.333	0.000	0.333	0.14
<b>Link1</b>						
FROM Router2	6	0	418.750	414.583	833.333	4.19
FROM Router1	9	0	556.944	390.873	833.333	8.35
Polling	3	0	800.000	0.000	800.000	4.00
Polling CTL	3	0	800.000	0.000	800.000	4.00
Aloha	8	0	408.199	400.055	810.000	5.36
Aloha CTL	6	0	273.333	372.410	800.000	2.69

**Link** The name of the link being reported.

**From Node Name** For full duplex links transmissions can occur simultaneously and in both directions. The name of the transmitting node is reported.

**Frames Delivered** The number of frames removed from the output buffer at the transmitting node on the link and subsequently placed in the input buffer of the receiving node. Frames that are in transmission when the report is produced (because of transmission delay and propagation delay) are not reported.

**Frames Resent** On a link, a framing error probability may be specified which causes statistically picked frames to be retransmitted as if they were in error. This feature is generally used to model noisy lines. The number of retransmitted frames is reported.

**Average Transmission Delay** Transmission delay is the time between when the frame (which may be part of a packet or contain several packets) is created at the input to the link and when the frame is delivered at the end of the link. It includes transmission, contention-resolution (for LANs), and propagation time.

**Standard Deviation of Transmission Delay** The standard deviation of the transmission delay for completed packet transmissions. Quoting a standard deviation should not be interpreted to mean that the

## 8. Reports

measured value is normally distributed about the average—it probably is not.

### Maximum Transmission Delay

The maximum delay observed for any packet across the link.

### Link Utilization

The transmission time for a frame is calculated from its size divided by the link speed. The link is in use for this time plus the propagation delay across the link for each frame. Utilization is then reported as the total usage time in the simulation run divided by the simulation run length.

## 8.7 Random Access Link Performance

CACI COMNET III

Thu Feb 24 11:42:31

PAGE 28

RANDOM ACCESS LINK PERFORMANCE  
REPLICATION NUMBER 1

LINK NAME	CSMA/CD	Aloha
ACCESS PROTOCOL	CSMA/CD	ALOHA
COLLISION EPISODES	0	0
COLLIDED FRAMES	0	0
NBR OF TRIES TO RESOLVE		
AVERAGE	0.00	0.00
STANDARD DEVIATION	0.00	0.00
MAXIMUM	0	0
NBR OF DEFERRALS	0	N/A
DEFERRAL DELAY (MS)		
AVERAGE	0.00	N/A
STANDARD DEVIATION	0.00	N/A
MAXIMUM	0.00	N/A
DEFERRAL QUEUE SIZE (FRAMES)		
AVERAGE	0.00	N/A
STANDARD DEVIATION	0.00	N/A
MAXIMUM	0	N/A
MULTIPLE COLLISION EPISODES		
NBR EPISODES	0	0
AVG PER EPISODE	0.00	0.00
MAX PER EPISODE	0	0

Link Name                   The name of the link being reported.

Access Protocol            Which protocol is being used on the link of CSMA, CSMA/CD or ALOHA.

Collision Episodes         How many times a collision occurred on the link, that is when 2 or more nodes try to transmit inside the same collision window.

Collided Frames            The total number of frames involved in collisions.

Average Number Of Retries To Resolve  
When a collision occurs, each colliding frame has to be retried at some later time. The retry frame may also collide. From the point of view of the first frame in the retry sequence, this figure reports the average number of retries attempted before successful transmission.

Standard Deviation Of Number Of Retries To Resolve  
The standard deviation of the number of retries to resolve for frame transmissions that have been completed. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average - it

## 8. Reports

probably is not.

### Maximum Number Of Retries To Resolve

During the simulation, the maximum observed number of retries that had to be attempted before an initially collided frame was transmitted.

### Number Of Deferrals

If a node attempts to transmit a frame and sees the link busy, it defers its transmission until the link becomes idle (plus the contention interval). This figure reports how many transmission attempts had to be deferred.

### Average Deferral Delay

The average delay due to transmission deferrals.

### Standard Deviation Of Deferral Delay

The standard deviation of the deferral delay for frame transmissions that have been completed. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average - it probably is not.

### Maximum Deferral Delay

The worst case deferral delay observed in the simulation.

### Average Deferral Queue Size(Frames)

The average number of frames queued in buffer because of deferral.

### Standard Deviation Of Deferral Queue Size(Frames)

The standard deviation of the deferral queue size. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average— it probably is not.

### Maximum Deferral Queue Size(Frames)

The maximum deferral queue size observed during the simulation.

### Number Of Multiple Collision Episodes

When a collision occurs in the collision window, it may be between two or more frames arising from different nodes. The simple case is between just two frames, but in a heavily congested system it is possible that frames from more than two nodes are involved in the collision. This figure reports the number of episodes where more than two frames are involved.

### Average Collisions Per Multiple Collision Episode

The average number of frames colliding in a multiple collision episode. This does not include collisions of the retry frames.

### Maximum Collisions Per Multiple Collision Episode

The maximum number of frames colliding in one contention interval for multiple collision episodes.

## 8.8 Link Frame Size Report

This report gives statistics on the frame sizes that were transmitted on the link.

CACI COMNET III RELEASE 1.2 BUILD 212 Thu Mar 07 12:54:05 PAGE 10

Acme2

LINKS: FRAME SIZE

REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

LINK	COUNT	FRAME SIZES (BYTES)		
		AVERAGE	STD DEV	MAXIMUM
LanB	713	51.365	42.549	144.000
LanDC	706	52.152	43.352	143.000
B-DC				
FROM RouterB	0	0.000	0.000	0.000
FROM RouterDC	0	0.000	0.000	0.000
B-NYC				
FROM RouterB	355	32.608	43.382	123.000
FROM RouterNYC	358	28.140	41.587	123.000
DC-NYC				
FROM RouterDC	357	31.796	43.151	122.000
FROM RouterNYC	349	30.493	43.547	122.000
LanNYC	1419	51.756	42.952	144.000

## 8. Reports

### 8.9 Link Utilization by Application

The Link Utilization by Application report provides a breakdown of link usage by application. For each category of packets flowing on a link, the report gives a count of the number of packets delivered, the average rate of delivery in kilobits per second (kbps), the percentage of bytes in the category, and the link utilization percentage produced by packets in the category.

A packet's application type is determined by the command (or source) that produced the packet.

### 8.10 Link Utilization by Protocol

The Link Utilization by Protocol report provides a breakdown of link usage by protocol. For each category of packets flowing on a link, the report gives a count of the number of packets delivered, the average rate of delivery in kilobits per second (kbps), the percentage of bytes in the category, and the link utilization percentage produced by packets in the category.

A packet's protocol is determined by the command (or source) that produced the packet.

## 8. Reports

### 8.11 Message Delays For Message & Response Sources

This report presents message delay statistics. For each originating node in the model it lists delays to each destination. Data for computer group nodes is aggregated. If individual node statistics are desired for the computer group nodes, simply set the “Show Group Node Detail” item when setting the report.

CACI COMNET III Thu Feb 24 11:42:26 PAGE 1

MESSAGE DELAYS FOR MESSAGE AND RESPONSE SOURCES  
REPLICATION NUMBER 1

ORIGIN / MSG SRC NAME: DESTINATION LIST	MESSAGES ASSEMBLED	MESSAGE DELAY (MILLISECONDS)		
		AVERAGE	STD DEV	MAXIMUM
StationATM / src RespATM: ECHO	15	11.487	0.000	11.487
Station1 / src MsgATM: StationATM	9	11.487	0.000	11.487

Origin/Message Source Name For each node that has message and/or response sources attached to it, list the name of the node and the name of the source.

Destination List The destination of the message. All destinations in the destination list of the source are listed. For Random Neighbor destinations, all nodes connected via one link to the originating node are listed.

Messages Assembled For each destination, the number of messages that have been completely assembled at the destination. Messages are broken into packets at the source node according to the transport protocol characteristics and then each packet is sent to the destination. Only messages where all packets have been received are reported.

Average Message Delay (milliseconds) The time between creating the first packet of the message on the originating node and the time of receiving the last packet of the message on the destination node, averaged over all messages sent during the simulation.

Standard Deviation Of Message Delay (milliseconds) The standard deviation of the message delay. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average—it probably is not.

Maximum Message Delay (milliseconds) The worst case message delay observed during the simulation.



## 8.12 Message Delays For Session Sources

CACI COMNET III

Thu Feb 24 11:42:26

PAGE 2

MESSAGE DELAYS FOR SESSION SOURCES  
REPLICATION NUMBER 1

ORIGIN / SESSION SRC: DESTINATION LIST	MESSAGES ASSEMBLED	MESSAGE DELAY (MILLISECONDS)		
		AVERAGE	STD DEV	MAXIMUM
Station1 / src SessATM: StationATM	6	11.487	0.000	11.487

**Origin/Session Source** For each node that has session sources attached to it, list the name of the node and the name of the source. Each source will send messages to the destinations listed in its destination list.

**Destination List** The destination of the message. All destinations in the destination list of the source are listed. For Random Neighbor destinations, all nodes connected via one link to the originating node are listed.

**Messages Assembled** For each destination, the number of messages that have been completely assembled at the destination. Messages are broken into packets at the source node according to the transport protocol characteristics and then each packet is sent to the destination. Only messages where all packets have been received are reported.

**Average Message Delay (milliseconds)** The time between creating the first packet of the message on the originating node and the time of receiving the last packet of the message on the destination node, averaged over all messages sent during the simulation.

**Standard Deviation Of Message Delay (milliseconds)** The standard deviation of the message delay. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average— it probably is not.

**Maximum Message Delay (milliseconds)** The worst case message delay observed during the simulation.

## 8. Reports

### 8.13 Message Delays For Transport & Answer Commands

CACI COMNET III

Thu Feb 24 11:42:26

PAGE 3

#### MESSAGE DELAYS FOR TRANSPORT AND ANSWER COMMANDS REPLICATION NUMBER 1

ORIGIN / COMMAND NAME:	MESSAGES	MESSAGE DELAY (MILLISECONDS)		
DESTINATION LIST	ASSEMBLED	AVERAGE	STD DEV	MAXIMUM
<hr/>				
Station2 / cmd Reply To Anyone:				
ECHO	0	0.000	0.000	0.000
Station2 / cmd Poller Message:				
Server3	1	3278.758	0.000	3278.758
Station3	2	2478.922	0.393	2479.316
Station3 / cmd Poll Reply:				
ECHO	2	2469.175	0.016	2469.175
Station4 / cmd Aloha Reply:				
ECHO	1	3279.175	0.000	3279.175

Origin/Command Name For each node that has transport commands or answer commands defined for it, list the name of the node and the name of the command.

Destination List The destination of the transport command. All destinations in the destination list of the command are listed. For Random Neighbor destinations, all nodes connected via one link to the originating node are listed.

Messages Assembled For each destination, the number of messages that have been completely assembled at the destination. Messages are broken into packets at the source node according to the transport protocol characteristics and then each packet is sent to the destination. Only messages where all packets have been received are reported.

Average Message Delay (milliseconds) The time between creating the first packet of the message on the originating node and the time of receiving the last packet of the message on the destination node, averaged over all messages sent during the simulation.

Standard Deviation Of Message Delay (milliseconds) The standard deviation of the message delay. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average—it probably is not.

Maximum Message Delay (milliseconds) The worst case message delay observed during the simulation.

## 8.14 Message Delays For Setup Commands

CACI COMNET III

Thu Feb 24 11:42:27

PAGE

4

MESSAGE DELAYS FOR SETUP COMMANDS  
REPLICATION NUMBER 1

ORIGIN / COMMAND NAME: DESTINATION LIST	MESSAGES ASSEMBLED	MESSAGE DELAY (MILLISECONDS)		
		AVERAGE	STD DEV	MAXIMUM
Station2 / cmd Aloha Session:				
Server4	2	3276.659	0.010	3276.659
Station4	1	4056.659	0.000	4056.659

**Origin/Setup Command Name** For each node that has setup commands defined for it, list the name of the node and the name of the setup command. The setup command, when executed, causes a session to be established to the destination and messages to be sent to it.

**Destination List** The destination of the messages. All destinations in the destination list of the setup command are listed. For Random Neighbor destinations, all nodes connected via one link to the originating node are listed.

**Messages Assembled** For each destination, the number of messages that have been completely assembled at the destination. Messages are broken into packets at the source node according to the transport protocol characteristics and then each packet is sent to the destination. Only messages where all packets have been received are reported.

**Average Message Delay (milliseconds)** The time between creating the first packet of the message on the originating node and the time of receiving the last packet of the message on the destination node, averaged over all messages sent during the simulation.

**Standard Deviation Of Message Delay (milliseconds)** The standard deviation of the message delay. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average—it probably is not.

**Maximum Message Delay (milliseconds)** The worst case message delay observed during the simulation.

## 8. Reports

### 8.15 Packet Statistics For Message & Response Sources

CACI COMNET III

Thu Feb 24 11:42:27

PAGE 5

#### PACKET STATISTICS FOR MESSAGE AND RESPONSE SOURCES REPLICATION NUMBER 1

ORIGIN: DESTINATION LIST	NUMBER OF PACKETS				PACKET DELAY (MS)	
	CREATED	DELIVERED	RESENT	DROPPED	AVERAGE	MAXIMUM
StationATM / src RespATM: ECHO	15	15	0	0	11.487	11.487
Station1 / src MsgATM: StationATM	9	9	0	0	11.487	11.487

**Origin** For each node that has message and/or response sources attached to it, list the name of the node and the name of the source.

**Destination List** The destination of the message. All destinations in the destination list of the source are listed. For Random Neighbor destinations, all nodes connected via one link to the originating node are listed.

**Packets Created** At the origin, how many packets have been created to send to the listed destination. Not all packets need to be delivered before they appear on the report.

**Packets Delivered** At the destination, how many packets have been received. This may differ from the number of packets created by the number of packets that are in transit at the instant the report is written.

**Packets Retransmitted** Packets may be retransmitted from the origin because they are blocked at some point en route to the destination. Blocking can occur because input or output buffers are full, or because a node or link on the route fails.

**Packets Dropped** When a node or link fails, the user can specify whether the transmission should be reattempted. If so, the retransmissions will be counted in the packets retransmitted field. If no retransmission is specified then the packet will be dropped and reported here.

**Average Packet Delay (milliseconds)** The time between creating a packet on the originating node and the time of receiving the packet at the destination node, averaged over all packets sent during the simulation.

**Maximum Packet Delay (milliseconds)** The worst case packet delay observed during the simulation.

## 8.16 Message Delivery Report

The message delivery report presents statistics on the delay before the message is reassembled by the destination. The message may still be worked on by the source after the message is delivered because of retransmissions, waiting for acks, or for modeling the close-sequence of the connection.

CACI COMNET III RELEASE 1.2 BUILD 212 Thu Mar 07 12:54:05 1996 PAGE 12

Acme2

MESSAGE + RESPONSE SOURCES: MESSAGE DELIVERED

REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

ORIGIN / MSG SRC NAME: DESTINATION LIST	MESSAGES ASSEMBLED	MESSAGE DELAY (MILLISECONDS)		
		AVERAGE	STD DEV	MAXIMUM
Boston1 / src MsgB1: NYC	1	3190.089	0.000	3190.089
DC1 / src MsgDC1: NYC	2	472.914	222.885	695.799
NYC / src Authorization: ECHO	228	3602.530	2104.942	8372.911
BostonATMs / src MsgBoston: NYC	122	1937.642	1184.235	4510.420
DCATMs / src MsgDC: NYC	115	1618.511	1047.468	4332.212

## 8. Reports

### 8.17 Transport Retransmission Report

This report presents retransmission statistics when blocked packets are retransmitted. It reports on quantities such as the number of times a packet has to be retransmitted, and the number of packets that are retransmitted when a blocked packet occurs.

## 8.18 Transport Window and Packet Interval Report

The message delivery report presents statistics on the delay before the message is reassembled by the destination. The message may still be worked on by the source after the message is delivered because of retransmissions, waiting for acks, or for modeling the close-sequence of the connection.

CACI COMNET III RELEASE 1.2 BUILD 212 Thu Mar 07 12:54:05 1996 PAGE 14

Acme2

MESSAGE + RESPONSE SOURCES: WINDOW STATS

REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

ORIGIN: DESTINATION LIST	WINDOW		NO. OF RESETS	CONG AVOID	PKT CREATION INTERVAL (MS)		
	AVG	MAX			AVG	MAX	STD DEV
Boston1 / src MsgB1:							
NYC	1.00	1	0	0	9	9	0
DC1 / src MsgDC1:							
NYC	1.00	1	0	0	10	10	0
NYC / src Authorization:							
ECHO	1.00	1	0	0	1792	4519	1099
BostonATMs / src MsgBoston:							
NYC	1.00	1	0	0	11	28	4
DCATMs / src MsgDC:							
NYC	1.00	1	0	0	11	29	4

## 8. Reports

### 8.19 Transport Timeout Report

This report presents the time-out timer statistics and is most relevant for enhanced sliding window or TCP/IP window protocols that adapt their time-out timers based on measured round-trip delays. The timer is only sampled when blocked packets are retransmitted.

CACI COMNET III RELEASE 1.2 BUILD 212 Thu Mar 07 12:54:05 1996 PAGE 16

Acme2

MESSAGE + RESPONSE SOURCES: TIMEOUT

REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

ORIGIN:	RETRANSMIT TIMEOUT (MS)		ROUND TRIP TIME (MS)	
DESTINATION LIST	AVG	MAX	AVG	MAX
Boston1 / src MsgB1:				
NYC	0.000	0.000	3491.010	3491.010
DC1 / src MsgDC1:				
NYC	0.000	0.000	803.835	1096.720
NYC / src Authorization:				
ECHO	0.000	0.000	2021.604	4780.000
BostonATMs / src MsgBoston:				
NYC	0.000	0.000	2225.514	4628.175
DCATMs / src MsgDC:				
NYC	0.000	0.000	1923.618	4763.133



## 8.20 Transport Ack Delay

This report complements the packet delay report to give statistics on the acks created and dropped as well as the delay for the ack. The ack delay is a round trip delay from when the packet started to when the ack returned. The time starts with the last packet that results in sending the ack.

CACI COMNET III RELEASE 1.2 BUILD 212 Thu Mar 07 12:54:05 1996 PAGE 17

Acme2

MESSAGE + RESPONSE SOURCES: ACK DELAY

REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

ORIGIN: DESTINATION LIST	NUMBER OF ACKS		ACK DELAY (MS)		
	CREATED	DROPPED	AVERAGE	MAXIMUM	STD DEV
Boston1 / src MsgB1:					
NYC	2	0	3356.010	3491.010	135.000
DC1 / src MsgDC1:					
NYC	4	0	653.835	1096.720	300.383
NYC / src Authorization:					
ECHO	456	0	1907.106	4780.000	1087.124
BostonATMs / src MsgBoston:					
NYC	244	0	2105.218	4628.175	1185.340
DCATMs / src MsgDC:					
NYC	230	0	1798.245	4763.133	1059.908

## 8. Reports

### 8.21 Transport Assembly Interval

This report provides statistics on the delay between when packets are assembled at the destination. The assembly interval can result from delay variation through the network (due to varying buffer delays) or it can result from congestion on the destination node. In cell-based networks, this measure is the cell-delay variation, a quality of service measure for these services.

## 8.22 Transport Burst Size

This report provides statistics on the burst size measured by the traffic policy on the protocol, when the traffic policy is present. It is the burst measurement used to determine the DE status of the packet, or for the ATM traffic policy, it is the burst measurement used to determine the conformance of the packets (whether the packets should be immediately dropped because they exceed their traffic contract).

CACI COMNET III RELEASE 1.2 BUILD 212 Thu Mar 07 12:54:05 1996 PAGE 19

Acme2

MESSAGE + RESPONSE SOURCES: BURST SIZE

REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

ORIGIN:	BURST	TRAF	BURST SIZE			
DESTINATION LIST	UNITS	TYPE	MIN	MAX	AVG	STD DEV
Boston1 / src MsgB1:						
NYC	kBits	FRLB	0	0	0	0
DC1 / src MsgDC1:						
NYC	kBits	FRLB	1	1	1	0
NYC / src Authorization:						
ECHO	kBits	FRLB	0	4	2	1
BostonATMs / src MsgBoston:						
NYC	kBits	FRLB	0	4	1	1
DCATMs / src MsgDC:						
NYC	kBits	FRLB	0	5	1	1

## 8. Reports

### 8.23 Transport Packet Flag Report

This report provides information about the flags set on the packet: the DE (discard eligibility) flag from the protocol's traffic policing algorithm, and the congestion flags set by the buffers when they are set up for FECN or BECN. The report also shows the number of packets blocked, separated by their DE status.

## 8.24 Transport Packet Size

This report provides two functions.

The first function is to report on the maximum packet and window size resulting from the socket constraints at either end. This is important because the sockets' packet or window size constraint can reduce the maximum protocol window or packet size.

The second function is to report on the packet sizes that were actually created. This part is most useful for monitoring the sizes for message that are smaller than the packet sizes or for monitoring the sizes from external traffic sources.

CACTI COMNET III RELEASE 1.2 BUILD 212 Thu Mar 07 12:56:14 1996 PAGE 39

Acme2

MESSAGE + RESPONSE SOURCES: PKT SIZE

REPLICATION 2 FROM 60.0 TO 120.0 SECONDS

ORIGIN: DESTINATION LIST	MAX PKT	MAX WINDOW	PACKET SIZE MIN	AVG	MAX
Boston1 / src MsgBl:					
NYC	128	1	63	75	86
DC1 / src MsgDC1:					
NYC	128	1	53	71	86
NYC / src Authorization:					
ECHO	128	1	50	83	120
BostonATMs / src MsgBoston:					
NYC	128	1	51	85	120
DCATMs / src MsgDC:					
NYC	128	1	51	86	119

## 8. Reports

### 8.25 Packet Delays For Session Sources

CACI COMNET III

Thu Feb 24 11:42:27

PAGE 6

PACKET DELAYS FOR SESSION SOURCES  
REPLICATION NUMBER 1

ORIGIN / SESSION SRC: DESTINATION LIST	NUMBER OF PACKETS				PACKET DELAY (MS)	
	CREATED	DELIVERED	RESENT	DROPPED	AVERAGE	MAXIMUM
Station1 / src SessATM: StationATM	6	6	0	0	11.487	11.487

- Origin/Session Source Name** For each node that has session sources attached to it, list the name of the node and the name of the source. A session source will create a setup packet to transmit to the destination to establish a route, followed by messages, once the connect packet has been received back from the destination.
- Destination List** The destination of the session. All destinations in the destination list of the source are listed. For Random Neighbor destinations, all nodes connected via one link to the originating node are listed.
- Packets Created** At the origin, how many packets have been created to send to the listed destination. Not all packets need to be delivered before they appear on the report.
- Packets Delivered** At the destination, how many packets have been received. This may differ from the number of packets created by the number of packets that are in transit at the instant the report is written.
- Packets Retransmitted** Packets may be retransmitted from the origin because they are blocked at some point en route to the destination. Blocking can occur because input or output buffers are full, or because a node or link on the route fails.
- Packets Dropped** When a node or link fails, the user can specify whether the transmission should be reattempted. If so the retransmissions will be counted in the packets retransmitted field. If no retransmission is specified, then the packet will be dropped and reported here.
- Average Packet Delay (milliseconds)** The time between creating a packet on the originating node and the time of receiving the packet at the destination node, averaged over all packets sent during the simulation.
- Maximum Packet Delay (milliseconds)** The worst case packet delay observed during the simulation.

## 8.26 Packet Delays For Transport & Answer Commands

CACI COMNET III

Thu Feb 24 11:42:27

PAGE 7

PACKET DELAYS FOR TRANSPORT AND ANSWER COMMANDS  
REPLICATION NUMBER 1

ORIGIN / COMMAND NAME: DESTINATION LIST	NUMBER OF PACKETS				PACKET DELAY (MS)	
	CREATED	DELIVERED	RESENT	DROPPED	AVERAGE	MAXIMUM
<b>Station2 / cmd Reply To Anyone:</b>						
ECHO	0	0	0	0	0.000	0.000
<b>Station2 / cmd Poller Message:</b>						
Server3	1	1	0	0	3277.758	3277.758
Station3	2	2	0	0	2477.922	2478.316
<b>Station3 / cmd Poll Reply:</b>						
ECHO	2	2	0	0	2469.175	2469.175
<b>Station4 / cmd Aloha Reply:</b>						
ECHO	1	1	0	0	3279.175	3279.175

**Origin/Command Name** For each node that has transport and/or answer commands, list the name of the node and the name of the command.

**Destination List** The destination of the command. All destinations in the destination list of the command are listed. For Random Neighbor destinations, all nodes connected via one link to the originating node are listed.

**Packets Created** At the origin, how many packets have been created to send to the listed destination. Not all packets need to be delivered before they appear on the report.

**Packets Delivered** At the destination how many packets have been received. This may differ from the number of packets created by the number of packets that are in transit at the instant the report is written.

**Packets Retransmitted** Packets may be retransmitted from the origin because they are blocked at some point en route to the destination. Blocking can occur because input or output buffers are full, or because a node or link on the route fails.

**Packets Dropped** When a node or link fails, the user can specify whether the transmission should be reattempted. If so the retransmissions will be counted in the packets retransmitted field. If no retransmission is specified then the packet will be dropped and reported here.

**Average Packet Delay (milliseconds)** The time between creating a packet on the originating node and the time of receiving the packet at the destination node, averaged over all packets sent during the simulation.

**Maximum Packet Delay (milliseconds)** The worst case packet delay observed during the simulation.

## 8. Reports

### 8.27 Packet Delays For Setup Commands

CACI COMNET III

Thu Feb 24 11:42:27

PAGE 8

PACKET DELAYS FOR SETUP COMMANDS  
REPLICATION NUMBER 1

ORIGIN / COMMAND NAME: DESTINATION LIST	NUMBER OF PACKETS				PACKET DELAY (MS)	
	CREATED	DELIVERED	RESENT	DROPPED	AVERAGE	MAXIMUM
<hr/>						
Station2 / cmd Aloha Session:						
Server4	2	2	0	0	3276.659	3276.659
Station4	1	1	0	0	4056.659	4056.659

Origin/Setup Command Name	For each node that has setup commands attached to it, list the name of the node and the name of the command. A setup command will create a setup packet to transmit to the destination to establish a route, followed by messages once the connect packet has been received back from the destination.
Destination List	The destination of the session. All destinations in the destination list of the source are listed. For Random Neighbor destinations, all nodes connected via one link to the originating node are listed.
Packets Created	At the origin, how many packets have been created to send to the listed destination. Not all packets need to be delivered before they appear on the report.
Packets Delivered	At the destination how many packets have been received. This may differ from the number of packets created, by the number of packets that are in transit at the instant the report is written.
Packets Retransmitted	Packets may be retransmitted from the origin because they are blocked at some point en route to the destination. Blocking can occur because input or output buffers are full, or because a node or link on the route fails.
Packets Dropped	When a node or link fails, the user can specify whether the transmission should be re-attempted. If so the retransmissions will be counted in the packets retransmitted field. If no retransmission is specified then the packet will be dropped and reported here.
Average Packet Delay (milliseconds)	The time between creating a packet on the originating node and the time of receiving the packet at the destination node, averaged over all packets sent during the simulation.
Maximum Packet Delay (milliseconds)	The worst case packet delay observed during the simulation.



## 8.28 Blocked Call Statistics

This report gives details on calls attempted, retried and blocked. Also included is information on the average and maximum hops needed to complete calls. For each origin and call name, a line is provided listing this information for each connected destination.

CACI COMNET III

Thu Feb 24 11:42:29

PAGE 20

BLOCKED CALL STATISTICS  
REPLICATION NUMBER 1

ORIGIN / CALL NAME: DESTINATION LIST	CALLS ATTEMPTD	CALLS RETRY	BLOCK PROB	HOPS AVG	HOPS MAX
Server1 / call Data Call: Server2	8	0	0.125	2.0	2
SUBTOTAL	8	0	0.125	2.0	2
Server1 (TOTAL)	8	0	0.125	2.0	2
<b>** T O T A L S **</b>	<b>8</b>	<b>0</b>	<b>0.125</b>	<b>2.0</b>	<b>2</b>

Origin/Call Name

For each node that has call sources attached to it, list the name of the node and the name of the source. A call source will establish circuit switched routes across the network to the destination and hold the bandwidth on the route for the call holding time.

Destination List

The destination of the call. All destinations in the destination list of the source are listed. For Random Neighbor destinations, all nodes connected via one link to the originating node are listed. (N.B. A call source may not be multicast).

Calls Attempted

The number of calls that have been attempted in terms of routing new call instances to their destination.

Calls Retried

A call routing attempt may fail because no route can be found where all links or all nodes have sufficient remaining, operational capacity. If the call cannot be routed, the user can specify whether a call should be retried later, or dropped. The number of retries is reported.

Calls Blocked

The number of calls blocked due to insufficient routing capacity (as outlined above) is reported.

Average Hops

Different call instances may use different routes to reach the destination. The average number of hops (or links) used across all routes is reported.

Maximum Hops

The longest route used in hops (or links) during the simulation for the particular call source/destination combination.

## 8. Reports

### 8.29 Disconnected Call Statistics

This report gives details on calls attempted, carried, disconnected and rerouted. For each origin and call name, a line is provided listing this information for each connected destination.

CACI COMNET III

Thu Feb 24 11:42:29

PAGE 21

DISCONNECTED CALL STATISTICS  
REPLICATION NUMBER 1

ORIGIN / CALL NAME: DESTINATION LIST	PRI	CALLS ATTEMPTD	CALLS CARRIED	CALLS DISCON	CALLS REROUT
Server1 / call Data Call:	1				
Server2		8	7	0	0
SUBTOTAL		8	7	0	0
Server1 (TOTAL)		8	7	0	0
** T O T A L S **		8	7	0	0

Origin/Call Name

For each node that has call sources attached to it, list the name of the node and the name of the source. A call source will establish circuit switched routes across the network to the destination and hold the bandwidth on the route for the call holding time.

Destination List

The destination of the call. All destinations in the destination list of the source are listed. For Random Neighbor destinations, all nodes connected via one link to the originating node are listed. (N.B. A call source may not be multicast).

Priority

Call sources have a priority which is used when preemptive operation has been specified. In this case, when a high priority call is routed and no route is available, the call will attempt to preempt a lower priority call on the full link.

Calls Attempted

The number of calls that have been attempted in terms of routing new call instances to their destination.

Calls Carried

The number of calls that have been successfully routed to their destination.

Calls Disconnected

The number of calls disconnected due to link or node failure.

Calls Rerouted

The number of disconnected calls that were successfully rerouted to their destination.

## 8.30 Preempted Call Statistics

CACI COMNET III

Thu Feb 24 11:42:29

PAGE 22

PREEMPTED CALL STATISTICS  
REPLICATION NUMBER 1

ORIGIN / CALL NAME: DESTINATION LIST	PRI	CALLS ATTEMPTED	CALLS CARRIED	CALLS PREEMPTED
Server1 / call Data Call:	1			
Server2		8	7	0
SUBTOTAL		8	7	0
Server1 (TOTAL)		8	7	0
** T O T A L S **		8	7	0

Origin/Call Name

For each node that has call sources attached to it, list the name of the node and the name of the source. A call source will establish circuit switched routes across the network to the destination and hold the bandwidth on the route for the call holding time.

Destination List

The destination of the call. All destinations in the destination list of the source are listed. For Random Neighbor destinations, all nodes connected via one link to the originating node are listed. (Note: A call source may not be multicast).

Priority

Call sources have a priority which is used when preemptive operation has been specified. In this case, when a high priority call is routed and no route is available, the call will attempt to preempt a lower priority call on the full link.

Calls Attempted

The number of calls that have been attempted in terms of routing new call instances to their destination.

Calls Carried

The number of calls that have been successfully routed to their destination.

Calls Disconnected

The number of calls disconnected due to link or node failure on some part of their route.

Calls Preempted

The number of calls disconnected due to preemption by a higher priority call which requires bandwidth on one (or more) of the nodes or links used by the preempted call. The preempting call does not have to be between the same origin and destination or following the same route as the preempted call. Even if there is only one common node or link between the calls then preemption may occur.

## 8. Reports

### 8.31 Call Statistics by Node

CACI COMNET III

Thu Feb 24 11:42:29

PAGE 24

CALL STATISTICS BY NODE							
REPLICATION NUMBER 1							
NODE NAME	CALLS	CALLS	CALLS		CALLS	CALLS	CALLS
	ATTEMPTD	BLOCK AVAIL	BLOCK TRAFF	BLOCK PROB	CARRIED	DISCON- NECTED	PRE-EMPTED
Server1	8	0	0	0.00	8	0	0
Server2	7	0	0	0.00	7	0	0
Router1	7	0	0	0.00	7	0	0
ATM	0	0	0	0.00	0	0	0
StationATM	0	0	0	0.00	0	0	0
Station1	0	0	0	0.00	0	0	0
Station2	0	0	0	0.00	0	0	0
Router2	0	0	0	0.00	0	0	0
Server3	0	0	0	0.00	0	0	0
Station3	0	0	0	0.00	0	0	0
Server4	0	0	0	0.00	0	0	0
Station4	0	0	0	0.00	0	0	0

Node Name	List the name of the node.
Calls Attempted	The number of calls that have been attempted in terms of routing new call instances to their destination from this node.
Calls Blocked - Availability	The number of calls that could not be routed over this node because this node had failed.
Calls Blocked - Traffic	The number of calls that could not be routed over this node because all the node capacity had been allocated to other calls.
Blocking Probability	Total Calls Blocked divided by Calls Attempted for calls at this node.
Calls Carried	The number of calls that have been successfully routed to their destination over this node.
Calls Disconnected	The number of calls disconnected due to this node failing.
Calls Preempted	The number of calls which carried at this node and which have been disconnected due to preemption by a higher priority call routing through this node. Preemption occurs when there is insufficient bandwidth available on the node to route the higher priority call. The preempting call does not have to be between the same origin and destination or following the same route as the preempted call. Even if there is only one common node or link between the calls then preemption may occur.

## 8.32 Call Statistics by Link

CALL STATISTICS BY LINK							
REPLICATION NUMBER 1							
LINK NAME	CALLS	CALLS	CALLS		CALLS	CALLS	CALLS
	ATTEMPTD	BLOCK AVAIL	BLOCK TRAFF	BLOCK PROB	CARRIED	DISCON- NECTED	PRE- EMPTED
Link3	7	0	0	0.00	7	0	0
Link2	7	0	0	0.00	7	0	0
LinkB	0	0	0	0.00	0	0	0
LinkA	0	0	0	0.00	0	0	0
Link1	0	0	0	0.00	0	0	0

Link Name	List the name of the link.
Calls Attempted	The number of calls that have been attempted in terms of routing call instances to their destination over this link. The origin and destination are not necessarily connected to this link—it may be an intermediate part of the route.
Calls Blocked - Availability	The number of calls that could not be routed over this link because this link had failed.
Calls Blocked - Traffic	The number of calls that could not be routed over this link because all its bandwidth had been allocated to other calls.
Blocking Probability	Total Calls Blocked divided by Calls Attempted, for calls blocked and carried on this link.
Calls Carried	The number of calls that have been successfully routed to their destination over this link.
Calls Disconnected	The number of calls carried on this link and subsequently disconnected due to failure of this link.
Calls Preempted	The number of calls which carried on this link and which have been disconnected due to preemption by a higher priority call which requires bandwidth on this link. The preempting call does not have to be between the same origin and destination or following the same route as the preempted call. Even if there is only one common node or link between the calls then preemption may occur.

## 8.33 Node Utilization Statistics For Calls

NODE UTILIZATION STATISTICS FOR CALLS  
REPLICATION NUMBER 1

NODE NAME	% NODE		BANDWIDTH USED (KBPS)			NODE UTIL %
	AVAIL	FAILS	AVERAGE	STD DEV	MAXIMUM	
Server1	100.00	0	5	3	10	0.05
Server2	100.00	0	5	3	8	0.05
Router1	100.00	0	5	3	8	0.05
ATM	100.00	0	0	0	0	0.00
StationATM	100.00	0	0	0	0	0.00
Station1	100.00	0	0	0	0	0.00
Station2	100.00	0	0	0	0	0.00
Router2	100.00	0	0	0	0	0.00
Server3	100.00	0	0	0	0	0.00
Station3	100.00	0	0	0	0	0.00
Server4	100.00	0	0	0	0	0.00
Station4	100.00	0	0	0	0	0.00

**Node Name** List the name of the node. Each node has a bandwidth capacity which it may use to carry call traffic.

**% Availability** A node is not available when it is in the failed state. The percentage of time available is reported. This is the Up Time divided by the Run Length.

**Node Failures** The number of failures that occurred in the simulation.

**Average Bandwidth Used (Kbps)** When calls are routed through the node, each call has a bandwidth requirement. The average total bandwidth requirement for all calls routed through the node is reported.

**Standard Deviation Of Bandwidth Used (Kbps)** The standard deviation of the bandwidth used. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average—it probably is not.

**Maximum Bandwidth Used (Kbps)** The maximum peak bandwidth used on the node during the simulation.

**Node Utilization** Average Bandwidth Used/Total Available Bandwidth expressed as a percentage.

### 8.34 Link Utilization Statistics for Calls

This report provides utilization rates and other statistics for each link's circuit switched call traffic.

CACI COMNET III

Thu Feb 24 11:42:29

PAGE 25

LINK UTILIZATION STATISTICS FOR CALLS  
REPLICATION NUMBER 1

LINK NAME	CIRCUIT	CIRCUIT	BANDWIDTH USED (KBPS)			CIRCUIT
	%	GROUP	AVERAGE	STD DEV	MAXIMUM	GROUP
	AVAIL	FAILS				UTIL %
Link3	100.00	0	5	3	8	56.93
Link2	100.00	0	5	3	8	56.93
LinkB	100.00	0	0	0	0	0.00
LinkA	100.00	0	0	0	0	0.00
Link1	100.00	0	0	0	0	0.00

Link Name	The name of the link. Each link has a bandwidth capacity which it may use to carry call traffic. This is not shared with packet switched traffic.
Availability %	A link is not available when it is in the failed state. The percentage of time available is reported. This is the Up Time divided by the Run Length.
Link Failures	The number of link failures that occurred in the simulation.
Average Bandwidth Used (Kbps)	When calls are routed over the link, each call has a bandwidth requirement. The average total bandwidth requirement for all calls routed over the link is reported.
Standard Deviation Of Bandwidth Used (Kbps)	The standard deviation of the bandwidth used. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average—it probably is not.
Maximum Bandwidth Used (Kbps)	The maximum peak bandwidth used on the link during the simulation.
Link Utilization	Average Bandwidth Used/Total Available Bandwidth expressed as a percentage.

## 8. Reports

### 8.35 Setup Delays For Session Sources

CACI COMNET III

Thu Feb 24 11:42:27

PAGE 11

#### SETUP DELAYS FOR SESSION SOURCES REPLICATION NUMBER 1

ORIGIN / SESSION SRC: DESTINATION LIST	SESSIONS SETUP	SETUP DELAY (MILLISECONDS)		
		AVERAGE	STD DEV	MAXIMUM
Station1 / src SessATM: StationATM	6	1.436	0.000	1.436

**Origin/Session Source Name** For each node that has session sources attached to it, the name of the node and the name of the source. A session source will create a setup packet to transmit to the destination to establish a route, followed by messages once the connect packet has been received back from the destination.

**Destination List** The destination of the session. All destinations in the destination list of the source are listed. For Random Neighbor destinations, all nodes connected via one link to the originating node are listed.

**Sessions Setup** When a session is started a session setup packet is sent to the destination and a session connect packet is returned. The session is then counted as setup This figure reports how many such setups occurred in the simulation.

**Average Setup Delay (millisecs)** The time difference between creating the session setup packet and receiving back the session connect packet.

**Standard Deviation Of Setup Delay (millisecs)** The standard deviation of the setup delay. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average—it probably is not.

**Maximum Setup Delay (millisecs)** The worst case session setup delay observed during the simulation.



## 8.36 Setup Delays For Setup Commands

CACI COMNET III

Thu Feb 24 11:42:27

PAGE 10

SETUP DELAYS FOR SETUP COMMANDS  
 REPLICATION NUMBER 1

ORIGIN / SETUP CMD: DESTINATION LIST	SESSIONS SETUP	SETUP DELAY (MILLISECONDS)		
		AVERAGE	STD DEV	MAXIMUM
Station2 / cmd Aloha Session:				
Server4	2	35.263	1.394	36.657
Station4	1	58.099	0.000	58.099

**Origin/Setup Command Name** For each node that has session setup commands attached to it, list the name of the node and the name of the setup command. A setup command will create a setup packet to transmit to the destination to establish a route, followed by messages once the connect packet has been received back from the destination.

**Destination List** The destination of the setup command. All destinations in the destination list of the setup command are listed. For Random Neighbor destinations, all nodes connected via one link to the originating node are listed.

**Sessions Setup** When a session is started by a setup command a session setup packet is sent to the destination and a session connect packet is returned. The session is then counted as setup. This figure reports how many setups occurred in the simulation.

**Average Setup Delay (milliseconds)** The time difference between creating the session setup packet and receiving back the session connect packet.

**Standard Deviation Of Setup Delay (milliseconds)** The standard deviation of the setup delay. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average—it probably is not.

**Maximum Setup Delay (milliseconds)** The worst case session setup delay observed during the simulation.

## 8.37 Sessions Setup by Node

SESSIONS SETUP BY NODE			
REPLICATION NUMBER 1			
NODE	SESSIONS IN PROGRESS		
	AVERAGE	STD DEV	MAXIMUM
Server1	0.00	0.00	0
Server2	0.00	0.00	0
Router1	0.00	0.00	0
ATM	0.00	0.00	0
StationATM	0.00	0.00	0
Station1	0.00	0.00	0
Station2	0.00	0.00	0
Router2	0.00	0.00	0
Server3	0.00	0.00	0
Station3	0.00	0.00	0
Server4	0.00	0.00	0
Station4	0.00	0.00	0

**Node Name** A node may originate sessions, carry session packets en route to their destination, or be the destination of the session. In all cases, a session is setup through the node to carry the session traffic.

Sessions setup across subnetworks where a connectionless routing algorithm is in operation are not reported, even if some of the session packets are routed through the node. This is because the complete session is not associated with the particular node.

**Average Sessions In Progress** The average number of sessions in progress on the node, irrespective of whether it is the origin, destination or switching node.

**Standard Deviation Of Sessions In Progress** The standard deviation of the sessions in progress. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average—it probably is not.

**Maximum Sessions In Progress** The maximum number of concurrent sessions in progress on this node.

## 8.38 Sessions Setup by Link

CACI COMNET III

Thu Feb 24 11:42:32

PAGE 31

SESSIONS SETUP BY LINK  
REPLICATION NUMBER 1

LINK	SESSIONS IN PROGRESS		
	AVERAGE	STD DEV	MAXIMUM
Link3	0.00	0.00	0
Link2	0.00	0.00	0
CSMA/CD	0.00	0.00	0
Ring	0.00	0.00	0
LinkB	0.00	0.00	0
LinkA	0.00	0.00	0
Link1	0.00	0.00	0
Polling	0.00	0.00	0
Aloha	0.00	0.00	0

**Link Name** A link carries the packets that comprise a session. If a session is setup across a subnetwork that uses connection oriented routing then all packets of the session will follow the same route. The number of sessions setup across the named link is reported.

Sessions setup across subnetworks where a connectionless routing algorithm is in operation are not reported, even if some of the session packets are routed across the link. This is because the complete session is not associated with the particular link.

**Average Sessions In Progress** The average number of sessions in progress across the link, irrespective of whether it is the origin, destination or switching node.

**Standard Deviation Of Sessions In Progress** The standard deviation of the sessions in progress. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average—it probably is not.

**Maximum Sessions In Progress** The maximum number of concurrent sessions in progress on this link.

### 8.39 Session Lengths by Setup Command

SESSION LENGTHS FOR SETUP COMMANDS					
REPLICATION NUMBER 1					
ORIGIN	SETUP CMD NAME	SESSIONS ENDED	SESSION LENGTH (SECONDS)		
			AVERAGE	STD DEV	MAXIMUM
Station2	Aloha Session	3	4.673	1.924	7.394

**Origin** The node name which originates the session by setup command.

**Setup Command Name** The name of the setup command.

**Sessions Ended** As the simulation executes, sessions are setup and messages transmitted across them. A session is counted as complete when the last message of a session has been received at the destination, and all packets for response messages have been received at the session origination node, and all ACKS for all packets have been received, and all pending message notices created by the receipt of the session messages have been cleared. When these conditions are met the session is complete and cleared. The number of sessions that have ended naturally in this way is reported.

**Average Session Length (seconds)** The time between the session setup packet being created and the end of the session (see above). The average for all sessions created by the session source is reported.

**Standard Deviation Of Session Length (seconds)** The standard deviation of the session length. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average—it probably is not.

**Maximum Sessions Length (seconds)** The maximum length of sessions that have completed.

## 8.40 Session Lengths by Session Source

CACI COMNET III

Thu Feb 24 11:42:32

PAGE 33

### SESSION LENGTHS FOR SESSION SOURCES REPLICATION NUMBER 1

ORIGIN	SESSION SRC NAME	SESSIONS ENDED	SESSION LENGTH (SECONDS)		
			AVERAGE	STD DEV	MAXIMUM
Station1	SessATM	6	0.024	0.000	0.024

**Origin** The node name which originates the session by a session source.

**Session Source Name** The name of the session source.

**Sessions Ended** As the simulation executes, sessions are setup and messages transmitted across them. A session is counted as complete when the last message of a session has been received at the destination, and all packets for response messages have been received at the session origination node, and all ACKS for all packets have been received, and all pending message notices created by the receipt of the session messages have been cleared. When these conditions are met the session is complete and cleared. The number of sessions that have ended naturally in this way is reported.

**Average Session Length (seconds)** The time between the session setup packet being created and the end of the session (see above). The average for all sessions created by the session source is reported.

**Standard Deviation Of Session Length (seconds)** The standard deviation of the session length. Quoting a standard deviation should not be interpreted to mean that the measured value is normally distributed about the average—it probably is not.

**Maximum Sessions Length (seconds)** The maximum length of sessions that have completed.

## 8. Reports

### 8.41 Session Blocking By Session Command

CACI COMNET III

Thu Feb 24 11:42:27

PAGE 13

SESSION STATISTICS FOR SETUP COMMANDS  
REPLICATION NUMBER 1

ORIGIN / SETUP COMMAND NAME: DESTINATION LIST	NUMBER OF SESSIONS					
	TRIED	SETUP	RETRY	BLOCK	DISCON	RERTD
Station2 / cmd Aloha Session:						
Server4	2	2	0	0	0	0
Station4	1	1	0	0	0	0

Origin	The node name which originates the session by a session command.
Setup Command Name	The name of the session setup command.
Sessions Tried	As the simulation executes applications which call the session setup command are scheduled. These then try to execute the command. The attempt to setup a particular session will succeed or fail depending upon network conditions. For instance, route availability inside hop and session limits, buffer availability, etc. The number of attempts to setup a session by respective setup command is reported.
Sessions Setup	The number of session setup attempts which succeeded.
Sessions Retried	If a setup attempt fails, the session may be retried later depending on the settings you have entered. The number of retries is reported.
Sessions Blocked	The session setup attempt may block due to insufficient buffer space on routing nodes, or no route being available inside the hop limit, or no route being available because at least 1 link on all routes is at its session limit, or nodes/links have failed and no route is available. The number of blocks is reported.
Sessions Disconnected	A session in progress may be disconnected because a node or link through which it is routed fails. The session may optionally be rerouted if this happens. the number of disconnections is reported.
Sessions Rerouted	The number of sessions rerouted following disconnection is reported.

## 8.42 Session Blocking By Session Source

CACI COMNET III

Thu Feb 24 11:42:27

PAGE 12

### SESSION SOURCE STATISTICS REPLICATION NUMBER 1

ORIGIN / SESSION SRC NAME: DESTINATION LIST	NUMBER OF SESSIONS					
	TRIED	SETUP	RETRY	BLOCK	DISCON	RERTD
Station1 / src SessATM: StationATM	6	6	0	0	0	0

Origin	The node name which schedules the session by session source.
Session Source Name	The name of the session source.
Sessions Tried	As the simulation executes scheduling conditions for the session sources are satisfied and so a session setup attempt is made. The attempt to setup a particular session will succeed or fail depending upon network conditions. For instance, route availability inside hop and session limits, buffer availability, etc. The number of attempts to setup a session by respective session source command is reported.
Sessions Setup	The number of session setup attempts which succeeded.
Sessions Retried	If a setup attempt fails, the session may be retried later depending on the settings you have entered. The number of retries is reported.
Sessions Blocked	The session setup attempt may block due to insufficient buffer space on routing nodes, or no route being available inside the hop limit, or no route being available because at least 1 link on all routes is at its session limit, or nodes/links have failed and no route is available. The number of blocks is reported.
Sessions Disconnected	A session in progress may be disconnected because a node or link through which it is routed fails. The session may optionally be rerouted if this happens. the number of disconnections is reported.
Sessions Rerouted	The number of sessions rerouted following disconnection is reported.

## 8.43 Buffer Input By Node

CACI COMNET III

Thu Feb 24 11:42:29

PAGE 18

INPUT BUFFER USE BY NODE  
REPLICATION NUMBER 1

NODE	PACKETS		BUFFER USE (BYTES)		
	ACCEPTED	BLOCKED	AVERAGE	STD DEV	MAXIMUM
Server1	42	0	0	0	1000
Server2	15	0	0	0	1000
Router1	15	0	0	0	1000
ATM	42	0	0	0	1000
StationATM	21	0	0	0	1000
Station1	21	0	0	0	1000
Station2	6	0	0	0	1000
Router2	15	0	0	0	1000
Server3	1	0	0	0	1000
Station3	3	0	0	0	1000
Server4	8	0	0	0	1000
Station4	2	0	0	0	1000

## Node Name

The name of the node receiving packets. The node is connected to links via an interface port. Each port has an input buffer, the size of which is defined on the port (edit the arc that connects the link to the node to see this). The amount of input buffer space on the node is the sum total of all the port input buffer spaces.

In addition, the node has an upper maximum on the total amount of input buffer space that can be is use at one time across all ports. This is defined on the node parameters screen.

When a packet is received, first the port input buffer is checked to see if there is space. If so, then the node input limit is checked to see if the node in total has space. If both tests succeed then the packet is received into the input buffer.

## Packets Accepted

The number of packets received into the node across all input ports on the node.

## Packets Blocked

The number of packets blocked across all input ports on the node due to insufficient buffer space (either port space or total node space).

## Buffer Use - Average

The average amount of buffer space in use across all input ports on the node.

## Buffer Use - Std Dev

The standard deviation about the average of buffer space in use across all input ports on the node.

## Buffer Use - Max

The maximum amount of buffer space in use across all input ports on the node observed in the replication.



## 8.44 Buffer Input By Port

CACI COMNET III

Thu Feb 24 11:42:28

PAGE 16

INPUT BUFFER USE BY PORT  
REPLICATION NUMBER 1

NODE: CONNECTED LINKS	PACKETS		BUFFER USE (BYTES)		
	ACCEPTED	BLOCKED	AVERAGE	STD DEV	MAXIMUM
Server1:					
CSMA/CD	21	0	0	0	1000
LinkA	21	0	0	0	1000
Link3	0	0	0	0	0
Server2:					
Link2	6	0	0	0	1000
Ring	9	0	0	0	1000
Router1:					
Link3	0	0	0	0	0
Link2	9	0	0	0	1000
Link1	6	0	0	0	1000
ATM:					
LinkA	21	0	0	0	1000
LinkB	21	0	0	0	1000
StationATM:					
LinkB	21	0	0	0	1000
Station1:					
CSMA/CD	21	0	0	0	1000
Station2:					
Ring	6	0	0	0	1000
Router2:					
Link1	9	0	0	0	1000
Polling	2	0	0	0	1000
Aloha	4	0	0	0	1000
Server3:					
Polling	1	0	0	0	1000
Station3:					
Polling	3	0	0	0	1000
Server4:					
Aloha	8	0	0	0	1000
Station4:					
Aloha	2	0	0	0	1000

Node Name

The name of the node receiving packets. The node is connected to each link via an interface port. Each port has an input buffer, the size of which is defined on the port (edit the arc that connects the link to the node to see this). The amount of input buffer space on the node is the sum total of all the port input buffer spaces.

In addition, the node has an upper maximum on the total amount of input buffer space that can be is use at one time across all ports. This is defined on the node parameters screen.

When a packet is received, first the port input buffer is checked to see if there is space. If so, then the node input limit is checked to see if the node in total has space. If both tests succeed then the packet is received into the input buffer.

Connected Links

The name of the link whose input port is being reported.

## 8. Reports

Packets Accepted	The number of packets received into the port input buffer.
Packets Blocked	The number of packets blocked at this port input buffer.
Buffer Use - Average	The average amount of buffer space in use at this port input buffer.
Buffer Use - Std Dev	The standard deviation about the average of buffer space in use at this port input buffer.
Buffer Use - Max	The maximum amount of buffer space in use at this port input buffer.

## 8.45 Buffer Output By Node

CACI COMNET III

Thu Feb 24 11:42:29

PAGE 19

OUTPUT BUFFER USE BY NODE  
REPLICATION NUMBER 1

NODE	PACKETS		BUFFER USE (BYTES)		
	ACCEPTED	BLOCKED	AVERAGE	STD DEV	MAXIMUM
Server1	42	0	1	35	1000
Server2	15	0	40	195	1000
Router1	15	0	0	0	1000
ATM	42	0	3	50	1000
StationATM	21	0	1	35	1000
Station1	21	0	0	0	1000
Station2	9	0	0	0	1000
Router2	15	0	0	19	1000
Server3	0	0	0	0	0
Station3	3	0	0	0	1000
Server4	6	0	0	0	1000
Station4	2	0	0	0	1000

**Node Name** The name of the node sending packets. The node is connected to links via an interface port. Each port has an output buffer, the size of which is defined on the port (edit the arc that connects the link to the node to see this). The amount of output buffer space on the node is the sum total of all the port output buffer spaces.

In addition, the node has an upper maximum on the total amount of output space that can be is use at one time across all ports. This is defined on the node parameters screen.

When a packet is routed across the node it must be placed into an output buffer port. First the port output buffer is checked to see if there is space. If so, then the node output limit is checked to see if the node in total has space. If both tests succeed then the packet is placed into the output buffer.

**Packets Accepted** The number of packets placed by the node it's output buffers across all output ports on the node.

**Packets Blocked** The number of packets blocked across all output ports on the node due to insufficient buffer space (either port space or total node space).

**Buffer Use - Average** The average amount of buffer space in use across all output ports on the node.

**Buffer Use - Std Dev** The standard deviation about the average of buffer space in use across all output ports on the node.

**Buffer Use - Max** The maximum amount of buffer space in use across all output ports on the node observed in the replication.

## 8. Reports

### 8.46 Buffer Output By Port

CACI COMNET III

Thu Feb 24 11:42:28

PAGE 17

OUTPUT BUFFER USE BY PORT  
REPLICATION NUMBER 1

NODE: CONNECTED LINKS	PACKETS		BUFFER USE (BYTES)		
	ACCEPTED	BLOCKED	AVERAGE	STD DEV	MAXIMUM
<b>Server1:</b>					
CSMA/CD	21	0	0	0	1000
LinkA	21	0	1	35	1000
Link3	0	0	0	0	0
<b>Server2:</b>					
Link2	9	0	40	195	1000
Ring	6	0	0	0	1000
<b>Router1:</b>					
Link3	0	0	0	0	0
Link2	6	0	0	0	1000
Link1	9	0	0	0	1000
<b>ATM:</b>					
LinkA	21	0	1	35	1000
LinkB	21	0	1	35	1000
<b>StationATM:</b>					
LinkB	21	0	1	35	1000
<b>Station1:</b>					
CSMA/CD	21	0	0	0	1000
<b>Station2:</b>					
Ring	9	0	0	0	1000
<b>Router2:</b>					
Link1	6	0	0	0	1000
Polling	3	0	0	19	1000
Aloha	6	0	0	0	1000
<b>Server3:</b>					
Polling	0	0	0	0	0
<b>Station3:</b>					
Polling	3	0	0	0	1000
<b>Server4:</b>					
Aloha	6	0	0	0	1000
<b>Station4:</b>					
Aloha	2	0	0	0	1000

#### Node Name

The name of the node sending packets. The node is connected to each link via an interface port. Each port has an output buffer, the size of which is defined on the port (edit the arc that connects the link to the node to see this). The amount of output buffer space on the node is the sum total of all the port output buffer spaces.

In addition, the node has an upper maximum on the total amount of input output space that can be is use at one time across all ports. This is defined on the node parameters screen.

When a packet is routed across the node it must be placed into an output buffer port. First the port output buffer is checked to see if there is space. If so, then the node output limit is checked to see if the node in total has space. If both tests succeed then the packet is placed into the output buffer.

## 8.46 Buffer Output By Port

Connected Links	The name of the link whose output port is being reported.
Packets Accepted	The number of packets routed into the port output buffer.
Packets Blocked	The number of packets blocked at the port output buffer.
Buffer Use - Average	The average amount of buffer space in use at the port output buffer.
Buffer Use - Std Dev	The standard deviation about the average of buffer space in use at the port output buffer.
Buffer Use - Max	The maximum amount of buffer space in use at the port output buffer.

## 8. Reports

### 8.47 Cloud Throughput

The cloud throughput report (“Frame Counts by VC”) provides throughput measures for each virtual circuit both in terms of the number of frames that the virtual circuit accepted or dropped and the number of kilobits accepted or dropped. These numbers are further divided between normal and DE frames. Figure 2.5.9a shows a sample throughput report.

```
CACI COMNET III  RELEASE  1.1          Sat Oct 15 15:35:35 1994          PAGE    1

          CLOUD VC FRAMES AND KILOBITS

          REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

Cloud:
VC:  Frames / Kilobits
      Accepted      Dropped
      Normal      DE      Normal      DE
-----
Frame Relay  (Total kilobits Transmitted = 976 )
VC1          Frm          66          0          0          0
             kb          528         0          0          0
VC2          Frm          0           0          0          0
             kb          0           0          0          0
VC3          Frm          56          0          0          0
             kb          448         0          0          0
VC4          Frm          0           0          0          0
             kb          0           0          0          0
```

The first line of data in the report identifies the cloud that includes the following virtual circuits, and it lists the total number kilobits that were successfully transmitted through the cloud during the simulation.

For each virtual circuit, there are two rows of data. The first row lists the frames and the second row lists the kilobits. For each row there are separate counts for normal and DE frames and for frames accepted and dropped. In these counts, frames that are dropped due the excess burst size being exceeded will be counted as DE frames. Also, the accepted frames may be dropped later when they arrive at a filled exit buffer.

## 8.48 Cloud VC Frame Delay and Burst Size

The "Frame Delay by VC" report presents the statistics for the frame delay and burst size for each virtual circuit in the cloud. The frame delay statistics are for all the frames that that cloud successfully delivers. The burst size is monitored for each frame that the virtual circuit accepts, and thus it will capture instantaneous peak values.

CACI COMNET III RELEASE 1.1 Sat Oct 15 15:35:35 1994 PAGE 2

### CLOUD VC FRAME DELAY

REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

Cloud: VC	Frame Delay(MS)			Burst Size (kb)	
	AVG	SDEV	MAX	AVG	MAX
Frame Relay					
VC1	302	42	413	16	40
VC2	0	0	0	0	0
VC3	304	52	444	14	32
VC4	0	0	0	0	0

## 8. Reports

### 8.49 Cloud Access Link

The "Access Link Stats" report presents statistics for each access link in the cloud. For each access link, two rows are printed: the first is for the entry link and the second is for the exit link.

CACTI COMNET III RELEASE 1.1 Sat Oct 15 15:35:35 1994 PAGE 3

#### CLOUD ACCESS LINKS

REPLICATION 1 FROM 0.0 TO 60.0 SECONDS

Cloud:		Frames		Buffer (kbits)			%Util
Access Link	(Entry) (Exit)	Accepted	Dropped	MAX	AVG	STD	
Frame Relay							
<>A	Entry	56	0	--	--	--	13.49
	Exit	0	0	0	0	0	0.00
<>B	Entry	66	0	--	--	--	16.04
	Exit	0	0	0	0	0	0.00
<>HQ	Entry	0	0	--	--	--	0.00
	Exit	122	0	2	0	0	29.40

The first columns count the frames accepted or dropped by the link. For entry links, this count reflects the frames that found a path to the required destination. If a frame arrives that must go to a destination for which no virtual circuit is defined and the cloud does not allow transmission through non-VCs, then that frame will count as being dropped by the entry access link even though that frame's transmission will have utilized the link. For exit links, the count reflects the number of frames accepted or blocked by the exit buffer. In cases where a low priority frame is initially accepted by the buffer and then preempted by a higher priority frame, then that low priority frame will count as being dropped and not accepted.

The middle columns present statistics on the buffer sizes. No buffer is modeled for the entry link (the buffering here is in the port of the connected node) and thus those values are blank.

The last column shows the link utilization in terms of percentage. The utilization is time-averaged over the replication, where at each event, the utilization is the number of busy circuits divided by the total number of circuits.



## 8.50 Cloud Access Buffer Policy Report

This report is similar to the buffer policy report on node buffers but it applies to the buffer policy on the cloud access exit (or egress) buffer.

## 8. Reports

### 8.51 Cloud Early/Partial Packet Discard Report

This report presents statistics on the early and partial packet discard algorithms when they are used on the cloud access link.

### 8.52 Global Traffic Command Reports

All of the reports that are available for local transport, answer, and set-up commands are also available for the global commands.

The reports work a little differently, however. The local commands may be used in several sources attached to the same node but the statistics are collected at one place (on the command) and thus this command aggregates the statistics across multiple application sources that use this source. This subtlety can affect message delay results by aggregating unrelated delays in the same report when the command is used on multiple sources and the command uses the "use original message" text option or the command is an answer command.

The global command however, is cloned so that each source that uses a global command has its own copy of the global command. Thus different sources on the same node using the same global command will have separate reports for these commands. However, if the source reuses the same global command, it will reuse the same cloned copy of the command and thus aggregate the statistics of these instances. This behavior is likely to be what is desired since the reused command is probably doing something similar in each instance so that the statistics are comparable and related.

## 8. Reports

### 8.53 Response And Answer Destinations

The response sources and answer commands have a checkbox to make using the catch-all "ECHO" an option. If the checkbox is off, then the statistics are reported separately for each destination responded to by the response or answer. This is useful because the destinations may have different levels of congestion or the destinations may have different paths so that the delays are not appropriately aggregated in the same statistic.

However, this detail comes at the cost of longer initialization times and more memory than the simple ECHO. When statistics are not required or when it is appropriate to aggregate the statistics over all possible destinations (when all such destinations are on the same LAN segment, for instance), it is advantageous to continue using the ECHO option to save memory and improve the speed of the simulation.

## 8.54 Snapshot Reports and Alarms

Snapshot reports convey statistical information about the state of a simulated model, very much like post-run reports. They are also turned on and off using a hierarchical list interface similar to that of post-run report selection. Snapshot reports differ from post-run reports in these key ways:

1. Snapshot reports may be turned on or off, not only before the simulation starts, but also while the simulation is running.
2. The information provided by a snapshot report consists only of the value of a measurement, such as link utilization, at an instant in time or over a brief interval of time. At the next instant or interval, new information is computed and presented, and the previous information is lost. You obtain this perishable information by watching the simulation, or by selecting the menu/toolbar option Take Snapshots. If you need the information to be collected and saved for later use, you should instead use a post-run report.
3. Alarms can be set on snapshot performance measures. Alarm conditions ARE recorded, and alarms have the potential to trigger model traffic. Alarms can also be configured to pause the simulation.

As mentioned above, snapshot reports may be observed in one of two ways:

1. For each class of layout objects, such as nodes, you may select one snapshot report to be displayed. The value of the snapshot will be displayed as a small number or set of numbers above the icons associated with the objects. Note that the default condition is that no snapshot reports are displayed for any objects.
2. At any point in the simulation, you may select a layout object and then select the menu/toolbar option Take Snapshots. A dialog will appear listing the current values of all the snapshot performance measures currently turned on for that object.

Some snapshot reports, such as session counts, are updated whenever the value changes. Most, however, are updated at one of two times: at the end of a user-specified interval, or when the user selects Take Snapshots. In the former case, the reported value is an average of the measurement over the interval. In the latter case, the reported value is a weighted combination of the current average (over only a partial interval) and the previously reported value. The measurement interval is specified separately for nodes and links on the backbone or subnet property dialog.

Some snapshot reports, such as link utilization, involve two numbers, and when these snapshots are selected for display, you might have difficulty determining which number has which meaning. The Take Snapshots option presents a more verbose report that is handy for associating the numbers with their meanings.

Each snapshot report may be set to *sound* an alarm when the value exceeds a specified threshold. When the alarm sounds, the icon of the offending object will change to a specified color, and a record of the alarm condition will be kept and later added to the post-run report file. When an alarm sounds on an object inside a subnet, transit network, or cloud, the subnet/transit net/cloud icon will be colored in the 2-D view, and will retain that color until the alarm *clears*, even if subsequent alarms of different colors sound.

Note that nodes and links can be configured to trigger model traffic upon the sounding or clearing of an alarm. This a very powerful, and very dangerous, feature. It is important to remember that, in combination with triggers, alarms have the ability to change model behavior, in which case they are more than mere instrumentation.

If you have an alarm set in your model, but you don't want to watch the simulation waiting for a color change, you can configure the alarm to force the simulation into single step mode when it sounds. To permit the simulation to continue running without further interruption, typically you must first reconfigure the alarm so it no longer forces single step mode, then use the Trace menu option to turn off single step.

As with post-run reports, your simulation will run slower when you have more snapshot reports turned on. A few snapshot reports are on by default, but these snapshots will have no effect on speed of simulation. One is the Disk Error snapshot report. It counts number of disk errors, such as attempting to read from a non-existent file. When the

## 8. Reports

number exceeds 1, an alarm sounds. For most models, a disk error indicates a mistake in the model.

## 9. Percentiles and Plots

### 9.1 Overview of Plots and Percentiles

Percentiles and plots can be obtained via any dialog box with a Statistics button. Before running the simulation, use the Statistics button to select the desired statistics; edit the desired statistic in the list to turn on the Collect Stats and Save Observations options. After running the simulation, use the Statistics button again to view and then plot the selected statistic. After bringing up the plot window by clicking on the plot button, use Plot on the menubar to obtain histograms, percentiles, or plots of smoothed data.

When plots of saved data are viewed after the simulation using the statistics buttons, it is possible to obtain any desired percentile for the saved data. For example, one could determine the 95th percentile for message delays from a particular message source, provided “save data” was requested for that source prior to the simulation.

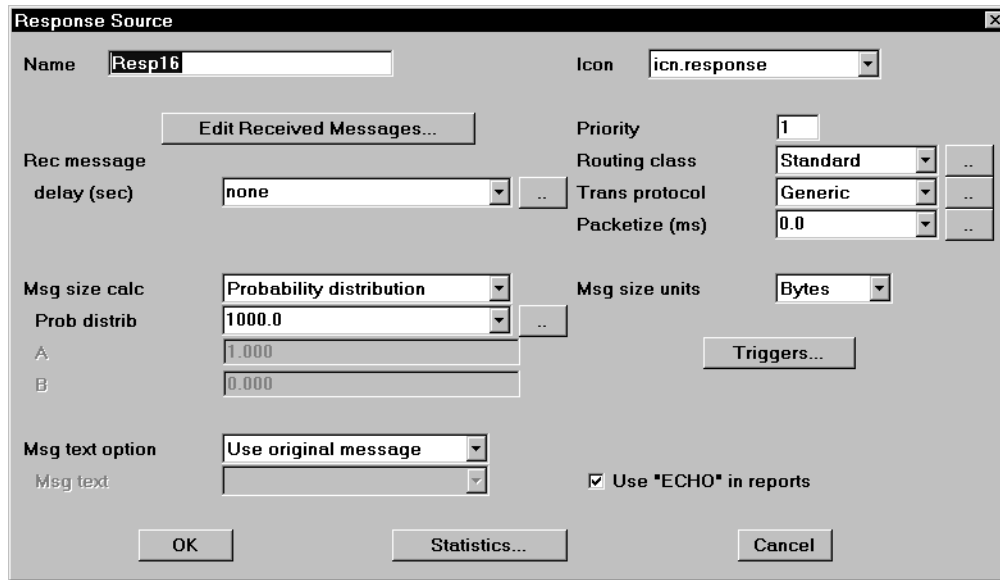
Post-processed plots of saved data allow the user to choose the number of points to be plotted, as well as the interval over which each plotted value is to be computed. For example, one could choose to plot 100 observations of utilization over an 80 second interval, with each observation of utilization computed as the time-weighted average utilization over the previous 2 second interval. If you request a plot of smoothed data for a delay type of statistic, you specify the time period  $[T1, T2]$  to be plotted and the number of points,  $N$ , to be plotted. The time period is divided into  $(T2 - T1) / N$  intervals and a single point is plotted for each interval. The point plotted for each interval is equal to the average of all of the delays observed during the interval. If no delays are observed, 0 is plotted.

If you request a plot of smoothed data for a level type of statistic (e.g., channel utilization), you specify the time period  $[T1, T2]$  to be plotted, the number of points,  $N$ , to be plotted, and the averaging interval,  $I$ , to be used in calculating the utilization at each plotted point. The point plotted at time  $T$  is the time-weighted average of the level statistic during the time interval  $[T-I, T]$ . If  $I > T$ , the time-weighted average is computed over the interval  $[0, T]$ . The averaging interval must be greater than or equal to  $(T2 - T1) / N$ . If the time period to be plotted is an entire replication and the averaging interval is set to the length of the replication, then each plotted point represents the time-weighted average of the level statistic from the beginning of the replication to the time of the plotted point.

### 9.2 Statistics Request Buttons

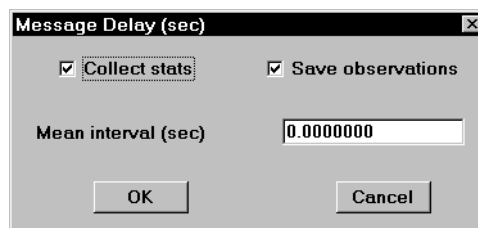
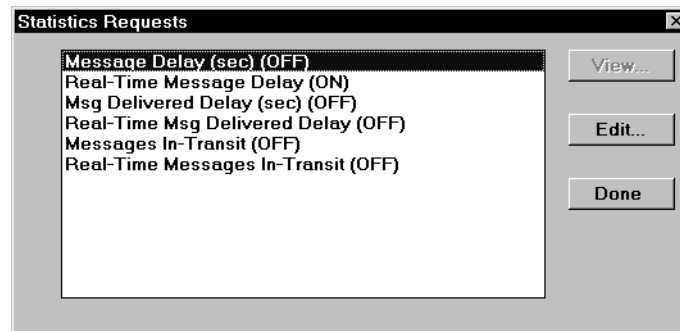
There are specific statistical monitors available for many of COMNET III's modeling elements. For example, links session sources, virtual circuits, access links, etc., all have statistics buttons on their dialog boxes. In order to collect statistics during a simulation, the desired statistics switches must be edited before starting the simulation.

## 9. Percentiles and Plots



Response Source Dialog Showing Statistics Button

The normal statistics are available at the end of the simulation: basic statistics such as mean, maximum, etc. are collected when the “collect stats” switch is turned on, and observations are collected for plotting after the termination of the simulation when the “save observations” switch is turned on. The normal statistics are generally available for both the cases where the statistics are updated for each event or where the statistics are updated with the mean values over an interval.

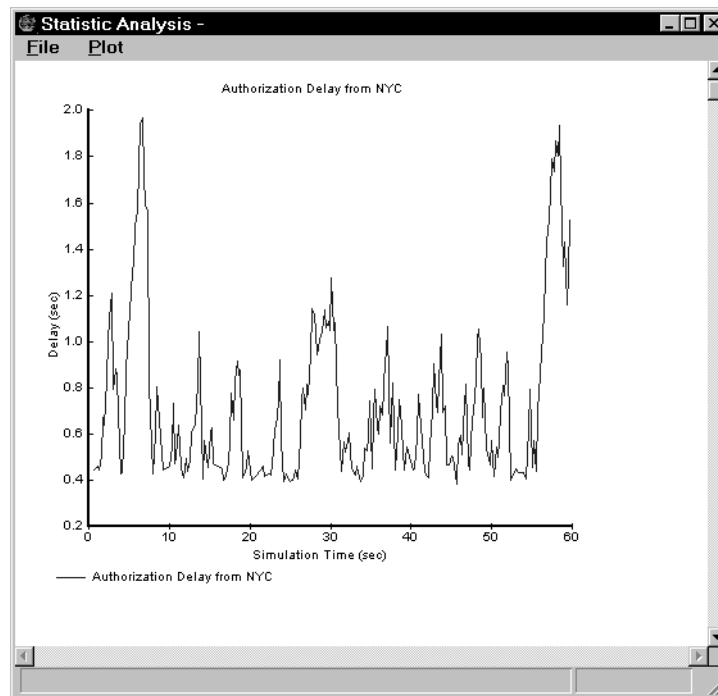


Turning on Message Delay Statistics for Post-simulation Plotting

In addition, the statistics averaged over the statistics interval may be set for real-time plotting, which is a plot that is



updated during the simulation for each update interval. These real-time plots must be turned on before starting the simulation through the edit button.



Post-simulation Plot

### 9.2.1 Access Link Statistics

The simulation can monitor statistics for each access link's entry and exit utilization, and the statistics for the buffer size at the exit end. The buffer size is the number of frame-level bytes (including frame overhead) that reside in the buffer for both the frames waiting for transmission on the exit link and the frames currently transmitting on the exit link. The link utilization is the ratio of the number of circuits busy transmitting frames over the total number of circuits available. Generally there will be just a single circuit and so the basic utilization will have either the value of zero or one.

The event statistics are labeled as "Exit Buffer Size," "Entry Link Utilization," and "Exit Link Utilization." Observations of these events may be saved for plotting or post-run analysis; however, there is an observation collected for each frame entering and leaving an access link and thus there will generally be a very large number of observations to record.

Averaged statistics for the buffer size and link utilization are also available and they are labeled as "Mean Exit Buffer Size," "Mean Entry Link Utilization," and "Mean Exit Link Utilization." These monitors will sample the averaged value at each sampling interval instead of each event, and thus there will be fewer observations to plot. However, the averaging interval can no longer be varied as part of the post-run analysis.

The average statistics may also be selected for real-time plots that will update at each update interval during the simulation. These real-time monitors are labeled "R-T Mean Exit Buffer Size," "R-T Mean Entry Utilization," and "R-T Mean Exit Utilization," where "R-T" is an abbreviation for real-time. (*The term "real-time" refers to the plots that are available and updating while the simulation progresses; it does not refer to whether the simulation itself is running in real-time.*)

## 9. Percentiles and Plots

### 9.2.2 Virtual Circuit Statistics

For virtual circuits, the simulation can measure the frame delay and the leaky-bucket burst size.

The frame delay is the time the frame takes to traverse the cloud from the moment it is created at the input to the entry link to the time it is delivered at the output of the exit link. This frame delay counts only the frames that successfully go through the cloud, and thus it does not count frames that are dropped within the cloud. The frame delay monitor labeled “Frame Delay” collects statistics for the delay for each frame that is delivered. The “Mean Frame Delay” monitors the averaged frame delay over the mean-statistics interval, and the “Real-Time Mean Frame Delay” provides a real-time plot of the averaged delay for this virtual circuit.

The burst size is the current size of the leaky-bucket burst at the time a frame is accepted by the virtual circuit and thus the frame size is added to the burst. The burst size contains only the data bytes for the encapsulated packets and thus it does not include the frame overhead or any padding necessary to meet minimum frame size. The monitor labeled “Burst Size” will collect the burst statistics for each frame accepted at the virtual circuit. The monitor labeled “Sampled Burst Size” collects burst size statistics based on a sample at the time of the mean-statistics update interval controlled by the cloud. “Real-Time Sampled Burst Size” provides a real-time plot of the sampled burst size.

Also, if the virtual circuit's “Show Burst Size” switch is on, then the sampled burst size for that virtual circuit will be displayed over the virtual circuit icon during the simulation.

### 9.3 Exporting Statistics Files

All of the simulation statistics collected during the file can be exported by clicking on File/Export/Simulation Statistics, which writes the simulation statistics to the file **statfile.xpt** in the model directory. Each line in the file has the following tab-delimited fields:

*Owner Type*—the type of object owning the statistic (e.g., PORT)

*Owner Name*—a name for fully identifying the object (e.g., for a PORT, the owner name is given by link:node:x, where x is I or O for input or output buffer).

*Statistic Type*—name of the type of statistic (e.g., possible values for a PORT are portbufferlevelI or portbufferlevelO).

*Replication*—the replication number; when there is more than one replication, replication -1 combines statistics for all reps as though the simulation consisted of 1 long rep; replication 0 treats each replication as one observation and provides the statistics based on these summary observations.

*Minimum*—the smallest observed value or level.

*Maximum*—the largest observed value or level.

*Sum*—for delay types of performance measures, the sum of the delays; for level statistics (e.g., buffer usage), the levels are time-weighted; to compute the mean for a level statistic, divide the Sum by the length of a replication.

*Sum of Squares*—for delay types of performance measures, the sum of the observed values squared; for level statistics, the levels are time-weighted.

*Count*—the number of observations.

# 10. SIMGRAPHICS II Graphics Editor

## 10.1 The SIMGRAPHICS II Graphics Editor

SIMDRAW, the SIMGRAPHICS II graphics editor, constructs images for animation, presentation graphics, and interactive graphical input. We will discuss here only those of its capabilities which are used to create or modify COMNET III icons and backgrounds.

The concept used by COMNET III and its SIMGRAPHICS II graphical interface is that graphic images, program menus and program dialog boxes are described using the SIMDRAW editor. These descriptions are stored in graphic library files which use the extension **.sg2**. The graphics images in these files can be tied to objects in a program, such as COMNET III.

COMNET III allows the user to provide new graphic images which can be stored in COMNET III's graphics libraries and used as icons and backgrounds in layouts.

Backgrounds can be added to layouts to give them more meaning or simply to improve their appearance. The icons can be associated with objects such as nodes, links and traffic sources.

Graphic images are built by drawing lines, circles, polygons, arcs, sectors. They can also include bitmaps images which are imported into SIMDRAW and added to graphic images. All of these graphics primitives can be grouped together to form complex images containing parts that can be manipulated independently.

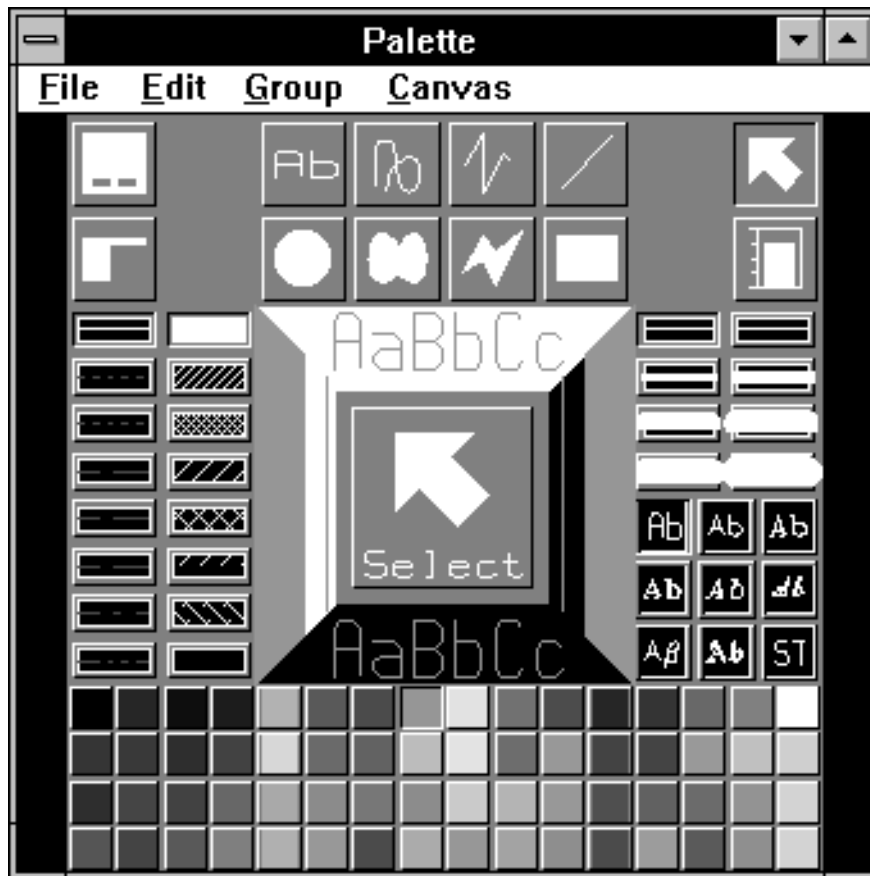
A collection of these images (as well as graphs, menus and dialog boxes) may be stored in a single library file. This file can be used with any language which supports SIMGRAPHICS II.

The graphics libraries delivered with COMNET III already contain icons for all standard program constructs as well as a selection of background images. The background image collection includes maps of various countries and continents against which network layouts can be constructed.

SIMDRAW is written entirely in MODSIM II using SIMGRAPHICS II.

## 10.2 Starting SIMDRAW

The SIMDRAW graphics editor is started by executing the program **simdraw**. On systems with graphical user interfaces, such as Windows, select the **simdraw** program icon to run it. After initialization three windows are displayed. The *Palette* window is shown here, the *Canvas* and *Tag* windows are initially empty.



Palette Window

### 10.3 Editor Windows

SIMDRAW uses a layout which is analogous to an artist's canvas and palette. Palette, canvas, and tag windows provide access to all of SIMDRAW's capabilities. All three windows are standard system windows which can be resized and moved.

The palette window contains colors, fill styles, line widths and modes for constructing objects. It also contains the menu bar which controls SIMDRAW. The menu bar allows access to additional capabilities not displayed on the palette.

The canvas window contains objects under construction. Images, graphs, and forms are built and displayed in the canvas window.

The tag window holds tags which represent groups of objects. Image groups, graphs, menu bars, and dialog boxes are represented by tags. For example, here is a tag window for a group of graphic objects:



Tag window showing tags for graphic images

## 10.4 Setting Modes, Styles, Colors and Line Widths

SIMDRAW has two general modes of operation, *create* and *select*. New objects can be constructed in create mode. In select mode, existing objects can be selected and modified.

SIMDRAW also maintains the current settings for color, fill styles, line styles, line widths, text fonts, and mark styles. These settings are used when objects are created or modified.

Most modes and settings are changed by clicking on the buttons in the palette window. There is a button associated with each type of object that can be built. There are also buttons associated with each color, line style, line width, and fill style. An indicator at the center of the palette window shows the current selections for the settings and mode. The current font and marker style are set by picking **Font Style** or **Marker Style** from the **Canvas** menu.

The circle and select buttons allow access to additional modes. Clicking on the circle button cycles between circle, arc, and sector modes. Clicking on the select button cycles between regular and extended select.

## 10.5 Constructing Graphic Images

Graphics are constructed by *dragging* or *clicking*. Dragging means to press the mouse button down, move the mouse to a new location, then release the button. While holding the mouse button down, a rubber band line or box tracks from the point where the button was pressed to the current mouse position. Clicking is also used to construct objects. Clicking means to press the mouse button down and release it. If the mouse has more than one button, the left-most button is used for clicking and dragging.

## 10. SIMGRAPHICS II Graphics Editor

All objects are constructed by picking a mode from the palette, then clicking or dragging in the canvas and palette windows.

Areas and polylines are built by clicking at the end point of each line segment. Lines are built by dragging from their starting point to their end point. Freehand polygons and polylines are constructed by dragging the mouse. As the mouse is moved an area or polyline is constructed automatically.

Sectors and arcs are built in two steps. First, a circle is drawn by defining a box. Next, a part of the circle is specified by dragging counterclockwise from the starting angle to the ending angle.

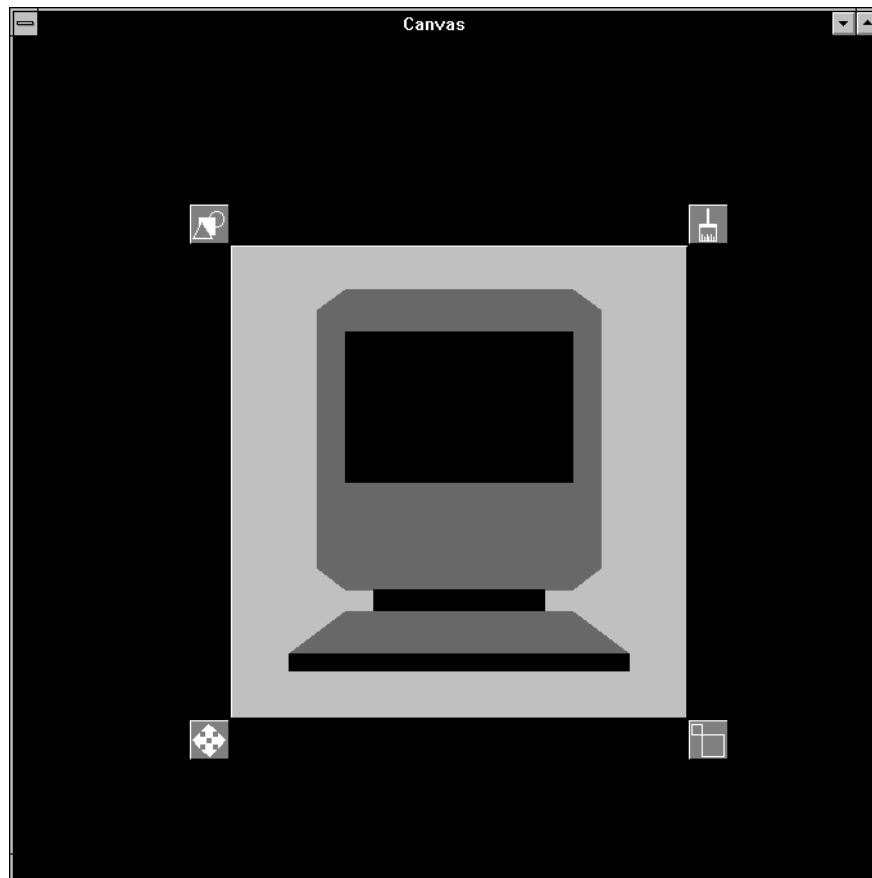
### 10.6 Selecting

To edit a graphic object it must first be selected.

Before a graphic object can be selected, SIMDRAW must be in the select mode. Selected objects can be modified or saved. The selected object has a box with four buttons surrounding it, or it is drawn in red. Clicking on an object selects it, and de-selects the previously selected object. Clicking on a tag in the tag window selects the group of objects the tag represents.

Some editor operations work on multiple objects. Extended select allows more than one object to be selected. Clicking on the select button until extended select is displayed places SIMDRAW in extended select mode.

In addition to clicking on a new object, selected objects can be de-selected in two ways. Clicking on a selected object de-selects it. In regular select mode, clicking on the background also de-selects it.



Canvas window showing selected icon

Clicking in the canvas window always selects the smallest selectable portion of an image or graph. To select a group of image primitives or a graph you must click on the tag in the tag window.

### 10.7 Moving, Painting, Resizing, and Changing Priority

A selected object that can be moved, painted, resized, or that can change priority has a box surrounding it with buttons which represent each of these operations. If an operation cannot be performed the corresponding button will be missing.

To paint an object, click on the paint button at the upper right corner of the object. Painting an object changes its attributes to match SIMDRAW's current settings. Color, fill style, line style, line width, text font and mark style may be changed by painting.

To change the size an image, drag the resize button at the lower right corner to the new size. An image can be flipped horizontally or vertically by dragging past its top or left sides.

An object's priority determines if it is on top of, or below other objects. Clicking on the priority button at the upper left corner changes its priority. If the object is below another object it is moved to the top, if it is on top it is moved below.

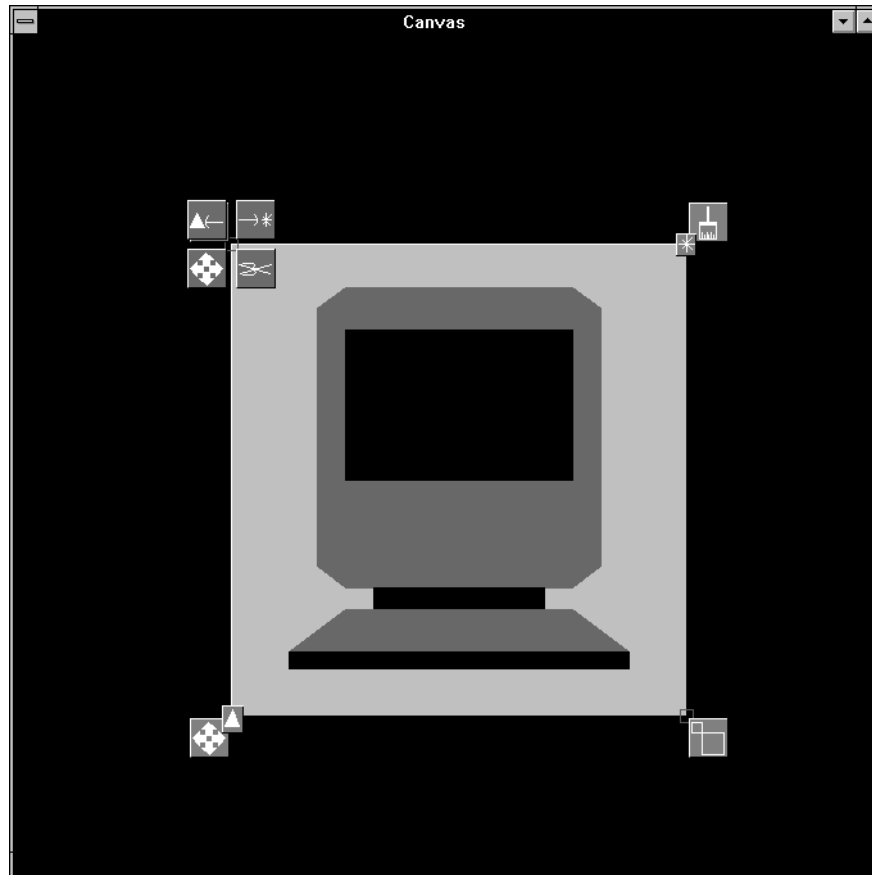
To move an object, drag the move button at the lower left corner to the desired location.

Note: for images which contain subgroupings, the subgroupings will change their priority in relation to other objects

## 10. SIMGRAPHICS II Graphics Editor

at the same level in the hierarchy only.

### 10.8 Editing Points



**Editing points**

If the selected object is an area or line, individual points may be moved, cut, or added. When an area or line is selected, all of its points are marked by squares. Clicking on a square selects the point and displays four buttons.

- Dragging the lower left button moves the selected point to a new location.
- Clicking on the lower right button removes the point.
- Clicking on the upper left button inserts a new point between the selected point and the point marked by a triangle.
- Clicking on the upper right button inserts a point between the selected point and the point marked by an asterisk.
- Clicking on the selected point de-selects it.

### 10.9 Cutting and Copying Objects

Selected objects can be removed and duplicated. Picking Cut from the Edit menu removes it. Picking Copy from the Edit menu duplicates it.

All objects in the canvas and tag windows can be removed by picking Clear from the Canvas menu.



## 10.10 Using the Grid

A grid can be used to perform precise positioning and sizing of images and graphs by breaking the canvas up into divisions. The grid is controlled by picking **Change Grid** from the **Canvas** menu.

If **Snap** is set, positioning and resizing is restrained to the intersections of the grid.

If **Grid** is set, lines are drawn on the canvas. The color of the grid lines can be set to the current color by picking **Change Grid Color** from the **Canvas** menu.

## 10.11 Canvas Dimensions and Coordinates

The dimensions of the canvas determine an object's coordinate system when it is saved. The dimensions should be set to match the world coordinate system used within the program. This ensures that the positions of objects set within SIMDRAW will remain the same when it is displayed within a program. The default coordinate system for SIMDRAW and COMNET III is (0...32767, 0...32767). It may be changed by picking **Dimension** from the **Canvas** menu.

The current location of the pointer relative to the canvas dimensions is displayed by picking **Show Coordinates** from the **Canvas** menu. Picking **Hide Coordinates** removes it. The coordinates are displayed in the same color as the grid at the lower left corner of the canvas window.

**Allow Icons to Scale** is an option available under **Canvas/Dimension**. If this option is set, an icon will automatically be scaled using the dimensions set in the editor, and the world coordinate system of the object it is attached to within the program. If it is not set, the icon will stay the same size no matter what world it is attached to.

## 10.12 The Library — Saving and Loading Objects

All objects are saved to and loaded from library files which have the extension **“.sg2”**. SIMDRAW always has a current library that objects are added to and removed from.

Selected objects can be saved by picking **Save** from the **File** menu. When an object is saved it is added to the current library and the entire library is written to a file. While in the save mode tags may be selected. This allows multiple objects to be saved before returning.

Objects may be loaded from a library file by picking **Load** from the **File** menu. Before an object can be loaded, the library containing the object must be loaded. Multiple objects may be loaded before returning.

Note: Individual image primitives cannot be saved. They must be grouped first.

## 10.13 Managing the Library

Since the library file can contain a collection of objects, individual objects may need to be renamed or removed. It may also be necessary to remove all of the objects in the library. These tasks can be performed by picking **Manage Library** from the **File** menu.

Individual objects are selected by clicking on their name in the list box provided. The selected object is renamed by entering a new name, and clicking on **Update**. It is removed by clicking on **Remove**. Clicking on **Remove All** removes all objects in the library.

Removing and renaming modifies the current working library only. The change to the library is not permanent until **Save To Library** is clicked on. You may undo any changes you have made by clicking on **Cancel**. If all of the objects are removed from the library and it is saved then the library file is deleted.

## 10.14 Grouping Images

Images can be grouped into complex hierarchies. SIMDRAW provides functions for creating, destroying, adding to, and removing from groups.

## 10. SIMGRAPHICS II Graphics Editor

Creating a group does not restrict the editing of the individual primitives. They can still be selected and modified. Selected image groups and primitives are grouped by picking **Create** from the **Group** menu. A tag is created which represents the new grouping. The selected group can be destroyed by picking **Destroy** from the **Group** menu.

After a group is created, additional primitives or groups may be added to it. Selected objects are added to a group by picking **Add To** from the **Group** menu, then selecting the tag representing the group to add to. Selected objects can be removed from their groups by picking **Remove From** from the **Group** menu.

Note: When creating a group, the selected objects cannot be members of any existing group.

### 10.15 Recentering Images

Each image or subimage has a center point which defines where the image will be drawn when it is placed on the screen. When an image is drawn at a (x,y) location its center point is aligned at (x,y). The center point is also used for rotation and scaling. The default center for an image is the middle of the smallest box which will enclose the image. The default center is set when an image or group is created or scaled.

Selected images are recentered by picking **Re-center** from the **Edit** menu. The current center of the image is displayed as a set of cross hairs. Clicking at a new location sets the center point.

### 10.16 Importing Text into SIMDRAW

Text files can be loaded into the graphics editor and added to an image. To do this, pick **Load** from the **File** menu. Type in the name of the text file as the name of the library, then press both the **Load Library** then **Load Object** buttons. A tag will appear in the tag window containing the text of the file.

### 10.17 Exiting SIMDRAW

Picking **Exit** from the **File** menu exits SIMDRAW. Any objects which have not been saved will be lost.

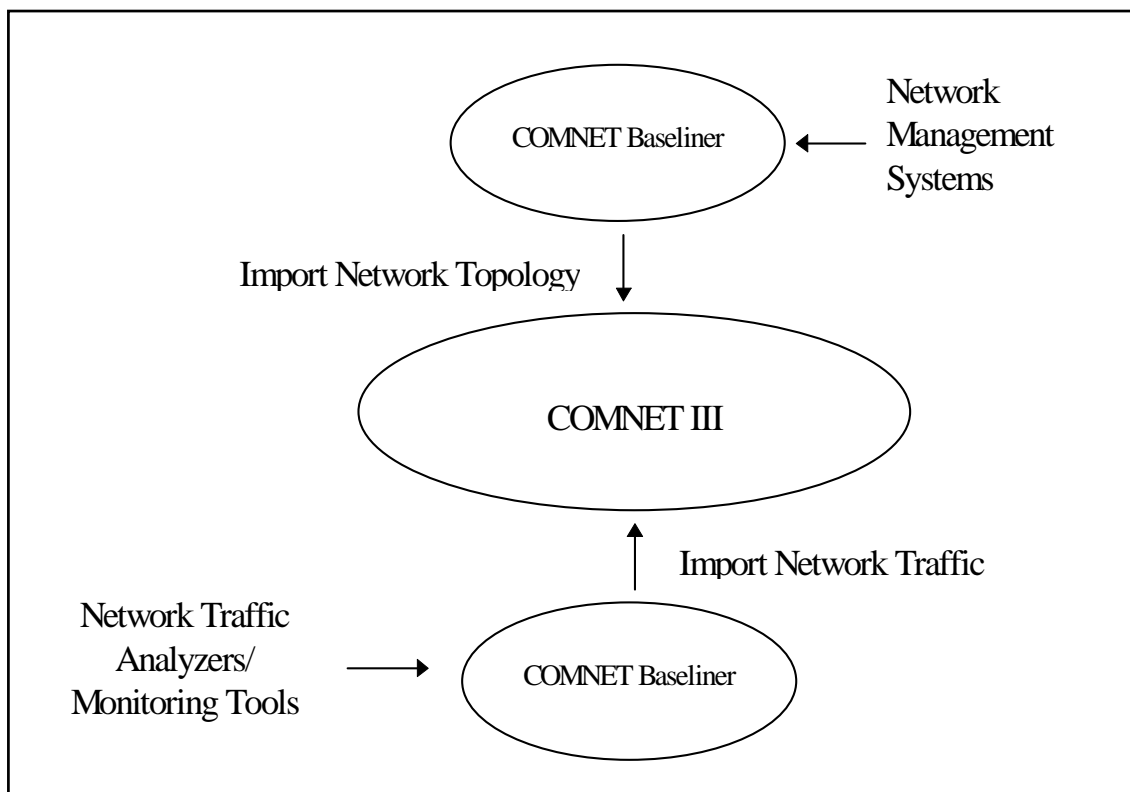
# 11. COMNET Baseline

## 11.1 INTRODUCTION

### 11.1.1 Overview

COMNET Baseline provides an interface between COMNET III and other networking tools. COMNET Baseline is a tool developed to assimilate data from your existing networking tools and automatically generate base-line COMNET III model information. Baseline model information reflects your existing network architecture and load characteristics. The automation process saves time which can be utilized on the important issues of design and planning. The automation process can run in either default mode, requiring the least intervention by the user, or in an advanced user mode, where the user can tailor the process to meet their specific needs. Typically, this process would be most useful during “what-if” scenarios of the model under study.

### 11.1.2 The System Architecture



COMNET Baseline helps you build a baseline model of your network by transforming data collected from a variety of network management and monitoring tools, into COMNET III model information. The model building process in COMNET III, see figure 1.2, can be split into two phases:

- a) building a network architecture model and,
- b) building a network load profile for the resulting model network.

## 11. COMNET Baseliner

### 11.1.3 Network Topology Information

The first step of building a COMNET III simulation model is to construct a topology of the network under study. COMNET Baseliner helps you build an accurate representation of your network architecture, by using the data captured by your existing network management tools. Currently supported Network Management Systems (NMS) include HP OpenView, Cabletron SPECTRUM, IBM NetView/6000 and Digital PolyCenter. It also provides an interface to basic SNMP tools such as Castlerock's SNMPc. COMNET Baseliner extracts node and link information from the data, in the topology file, to automatically generate a COMNET III model file. Based on the data available, COMNET Baseliner can identify different types of links, i.e. Ethernet, Token Ring, FDDI Ring or Point-to-Point (WAN) link. The import process may be tailored to suit your need in the advanced user mode. The node and link parameters, once in the COMNET environment, can be modified as needed.

### 11.1.4 Network Load Characterization

To represent your network further, COMNET Baseliner also allows you to add "real" traffic to your network architecture model. This is achieved by transforming traffic captured by network analyzers or network monitoring tools into load characteristics for COMNET III. COMNET III provides interface to external traffic collected at various levels of detail. At the highest level, traffic data may be collected at the application-layer level containing data such as packet counts and bytes counts. Or it may be collected at the frame-level detail, containing information on individual packets.

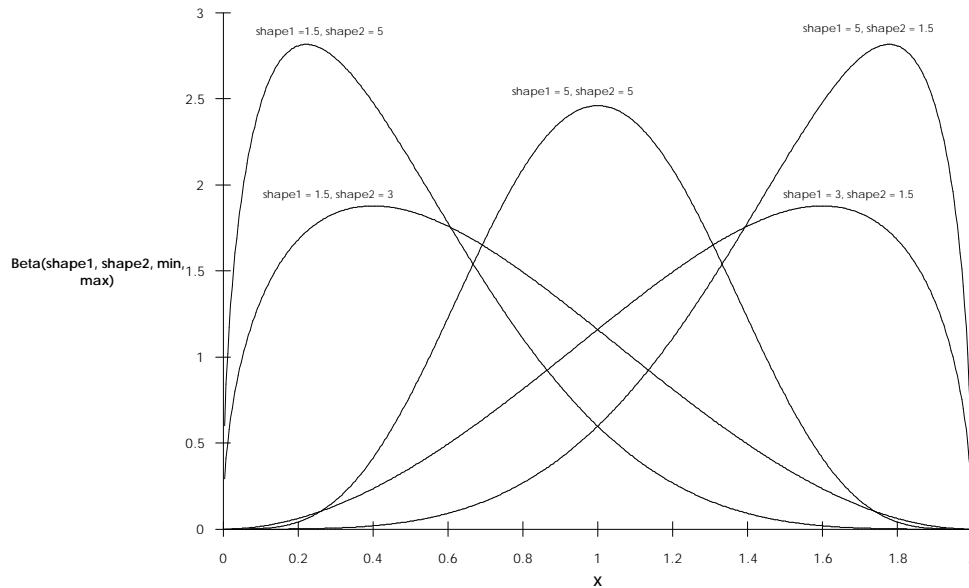
With conversation pair traffic, representing the highest level of detail, the import process uses the interrupted poisson process for characterizing the traffic flow. In the scenario, where the import process does not have enough data available, it will use the default poisson process for characterizing the traffic. On the other end of the spectrum, such as event trace traffic, the events recorded in the data file govern the traffic flow during simulation. The type of problem under study would determine what kind of data to import. External traffic can include traffic from multiple sources being sent over multiple segments.

External traffic may be mixed with internal traffic within the model. Within the simulation, external traffic behaves similar to internal traffic. This is particularly useful for trying out multiple "what-if" scenarios.

For a more detailed explanation of COMNET Baseliner and its features, please consult the separate COMNET Baseliner Users Manual.

# 12. Statistical Distribution Functions

## 12.1 The Beta Distribution



The Beta distribution function takes two positive real parameters, **shape1** and **shape2** which provide information on the shape of the probability distribution. It returns a positive, real number. The stream parameter specifies which random number stream will be used to provide the sample.

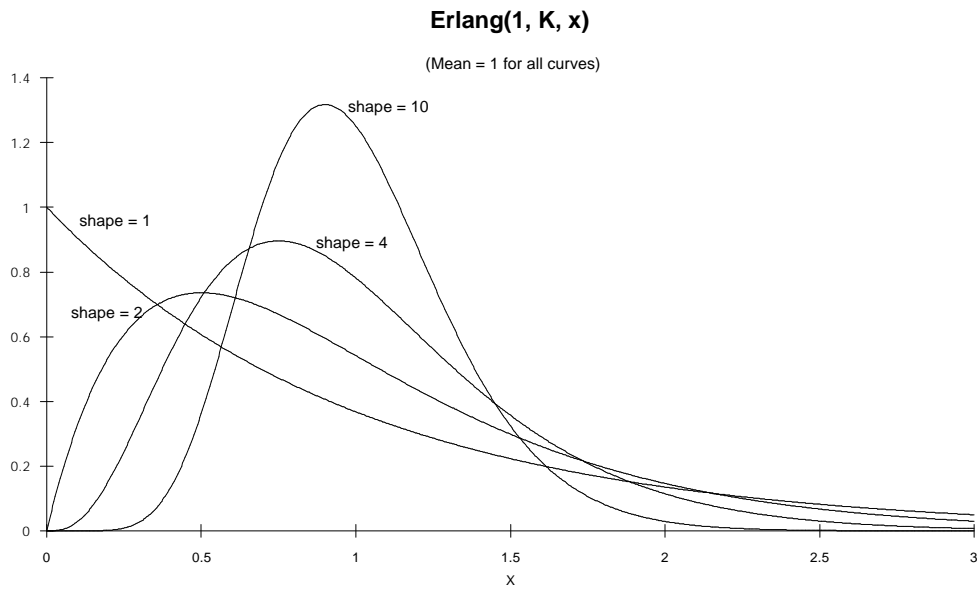
Beta(shape1, shape2, min, max, stream)

This is one of the most versatile distribution functions. It can take on a variety of shapes depending on the two shaping parameters. Normally its return,  $x$ , falls in the interval  $0 < x < 1$ , but COMNET III allows the distribution to be mapped over the interval specified by **min** and **max**. Note that the distribution plots above use the statistical terminology **shape1** and **shape2** for the two shaping parameters.

## 12. Statistical Distribution Functions

### 12.2 The Erlang Distribution

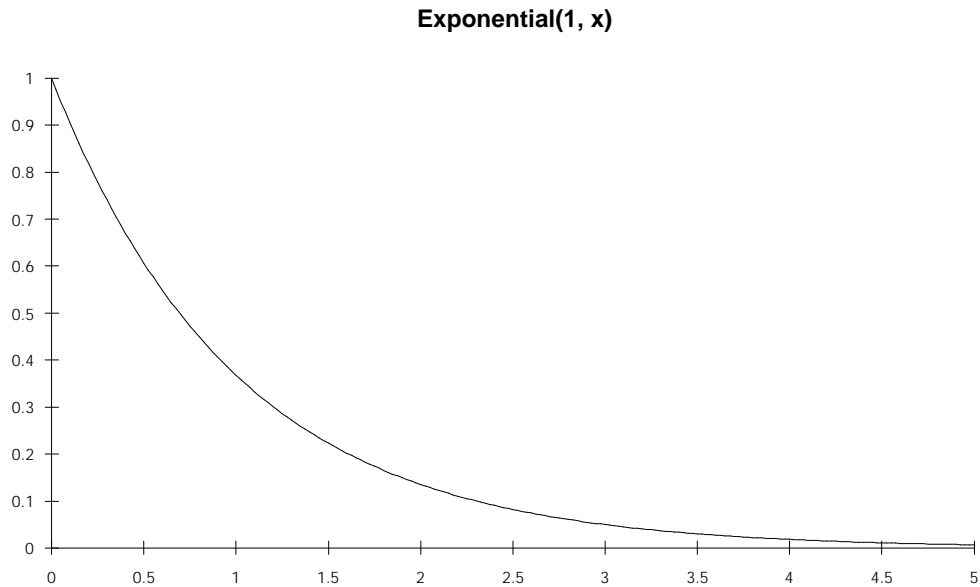
The Erlang distribution function takes two parameters, **mean** is a positive real number and **shape** is a positive integer. It returns a positive, real number. The **stream** parameter specifies which random number stream will be used to provide the sample. The larger the shape parameter, the sharper the peak in the distribution, and the less variance in samples. This distribution is named after A. K. Erlang, a Danish mathematician, who pioneered in the field of telecommunications analysis.



Erlang (mean, shape, stream)

## 12.3 The Exponential Distribution

The Exponential distribution function takes one positive, real number which is the **mean**. It returns a positive real number. The stream parameter specifies which random number stream will be used to provide the sample.



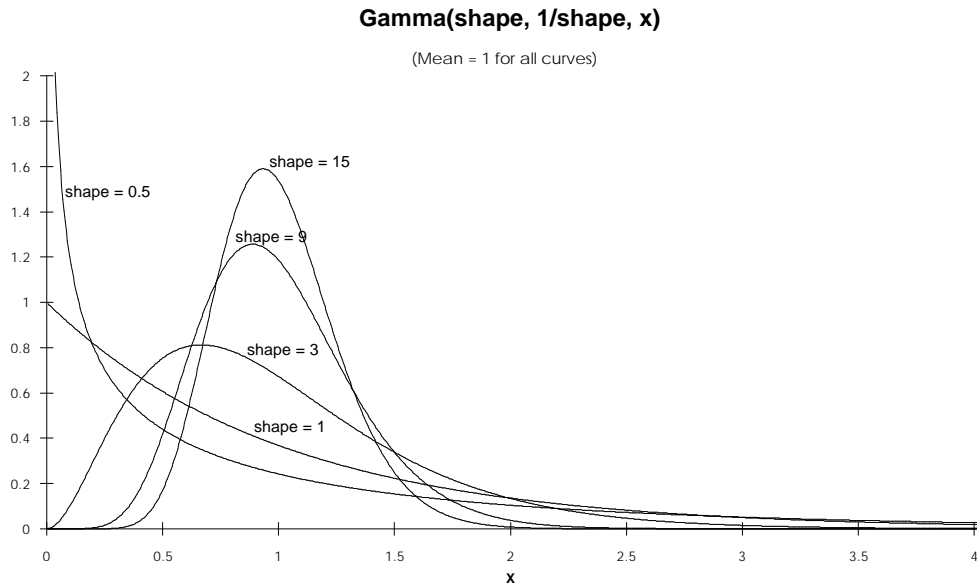
Exponential (mean, stream)

This distribution is widely used to model arrival times of events which follow a Poisson pattern. Each sample chosen from the Exponential function specifies the time which will elapse before the next arrival. This is called the interarrival time. Samples have a high probability of being less than the mean. This implies that the distribution has a long tail and will occasionally provide a sample significantly higher than the mean. This behavior is very useful in modeling random arrival patterns.

Although arrival patterns may be characterized as “Poisson,” the Poisson function takes as an argument a time interval and returns an integer which specifies how many arrivals will occur in that interval. In modeling it is more useful to know how much time will elapse before the next arrival. This is why the Exponential function is used to actually implement Poisson arrival patterns.

## 12. Statistical Distribution Functions

### 12.4 The Gamma Distribution



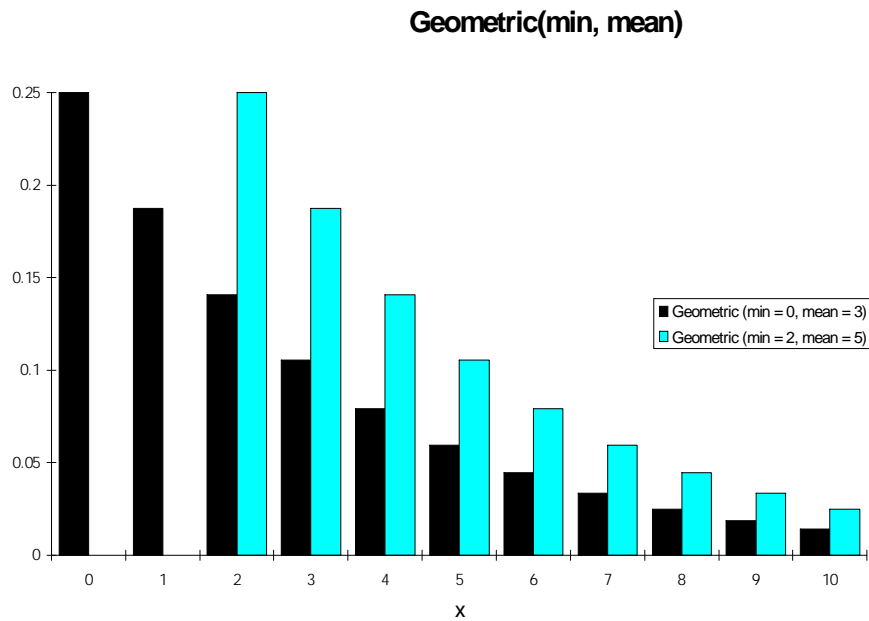
Gamma (mean, shape, stream)

The Gamma distribution function takes two positive, real numbers mean and shape which is a shaping parameter. It returns a positive real number. The stream parameter specifies which random number stream will be used to provide the sample.



## 12.5 The Geometric Distribution

The Geometric distribution function takes two positive numbers: min and mean; min must be an integer. The min value offsets the distribution from its normal minimum at zero. It returns a positive number. The stream parameter specifies which random number stream will be used to provide the sample.



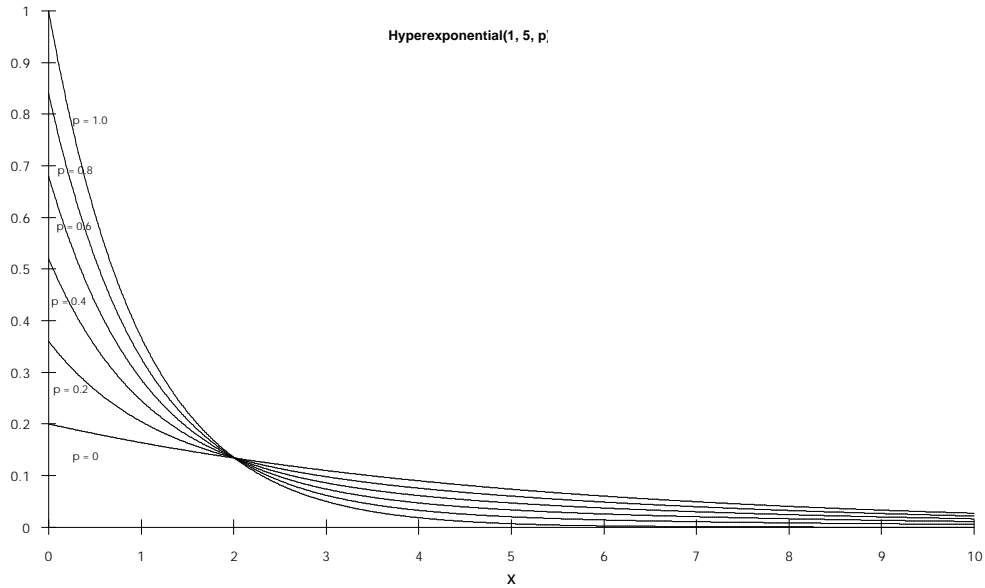
Geometric (min, mean, stream)

A discrete version of the exponential distribution.

## 12. Statistical Distribution Functions

### 12.6 The Hyperexponential Distribution

The Hyperexponential distribution function takes three positive, real numbers: mean1, mean2 and probability of mean. It returns a positive, real number. The stream parameter specifies which random number stream will be used to provide the sample.

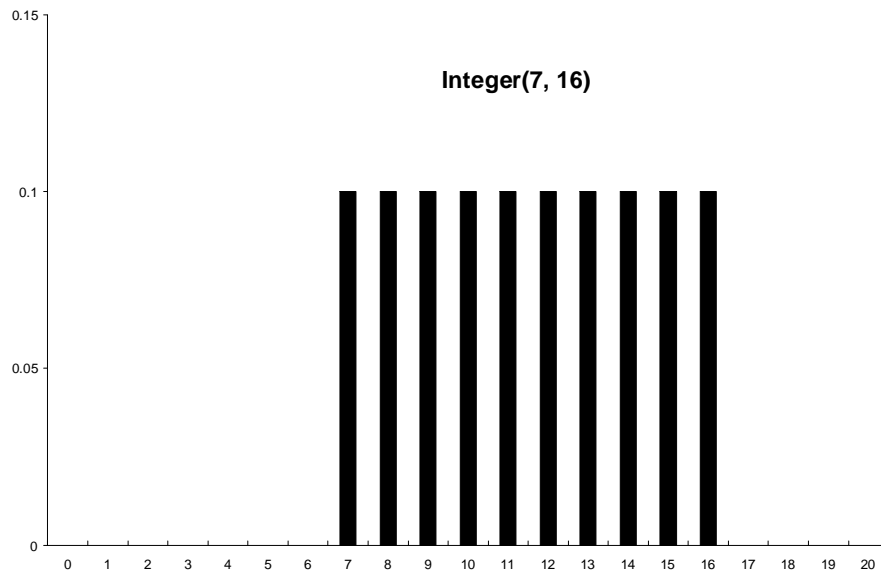


Hyperexponential(mean1, mean2, probability of mean, stream)

Possibly used for modelling the packet interarrival times for talk spurts in packetized voice networks.

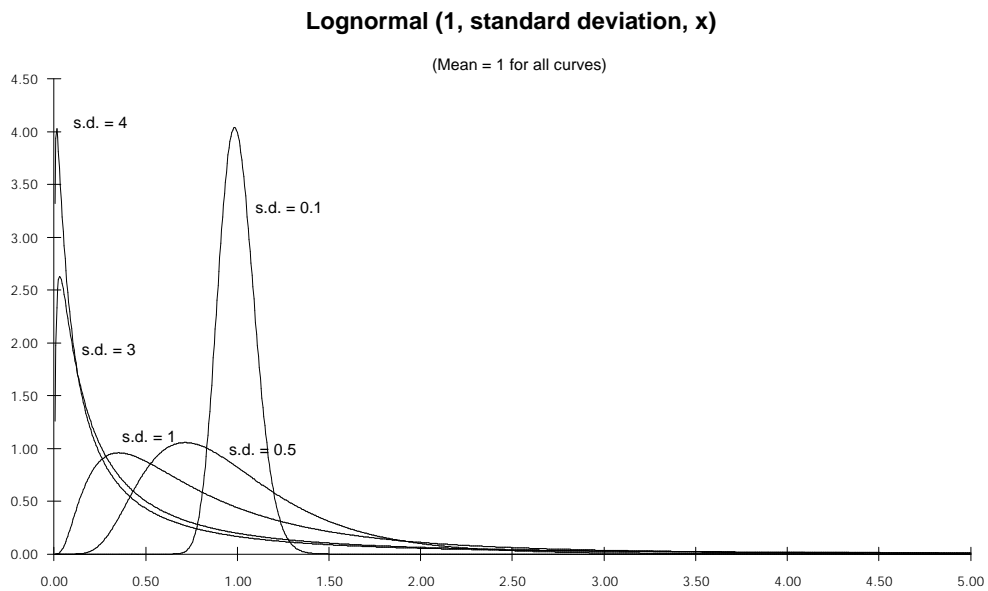
## 12.7 The Integer Distribution

The Integer distribution function takes two positive, real numbers: lower bound & upper bound. It returns a positive, real integer between these bounds inclusive. The stream parameter specifies which random number stream will be used to provide the sample.



## 12. Statistical Distribution Functions

### 12.8 The Lognormal Distribution

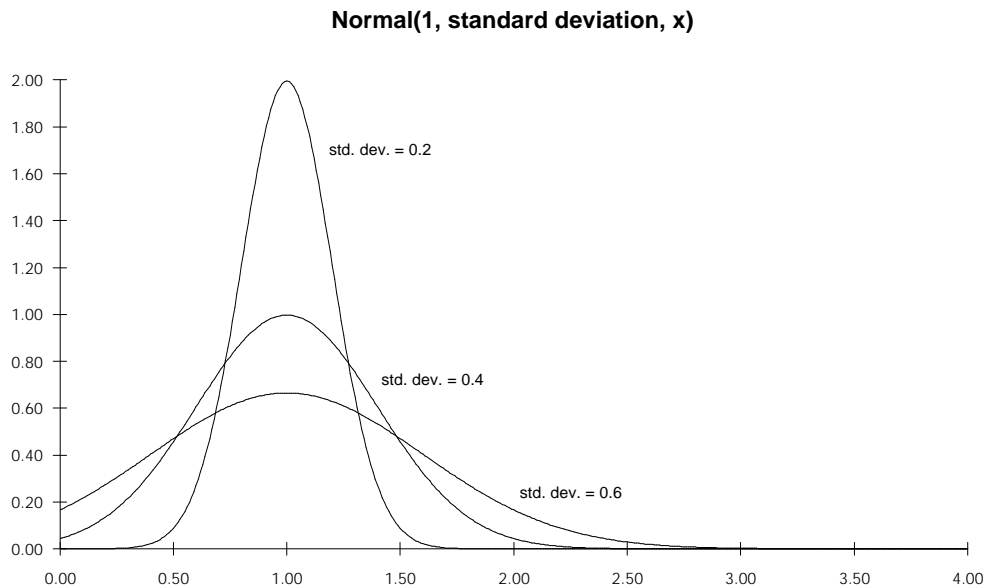


The Lognormal distribution function takes two positive, real numbers: mean and standard deviation. It returns a positive, real number. The stream parameter specifies which random number stream will be used to provide the sample.

LogNormal (mean, standard deviation, stream)

## 12.9 The Normal Distribution

The Normal distribution function takes two positive, real numbers: mean and standard deviation. It returns a positive, real number. The stream parameter specifies which random number stream will be used to provide the sample. In COMNET III this function has been modified so that it will not return a negative number. In COMNET III the Normal distribution is truncated so that it does not produce negative numbers. If the mean you choose is more than about three times the standard deviation this will have little effect, since there will only be a very small portion of the Normal distribution to the left of the origin anyway.

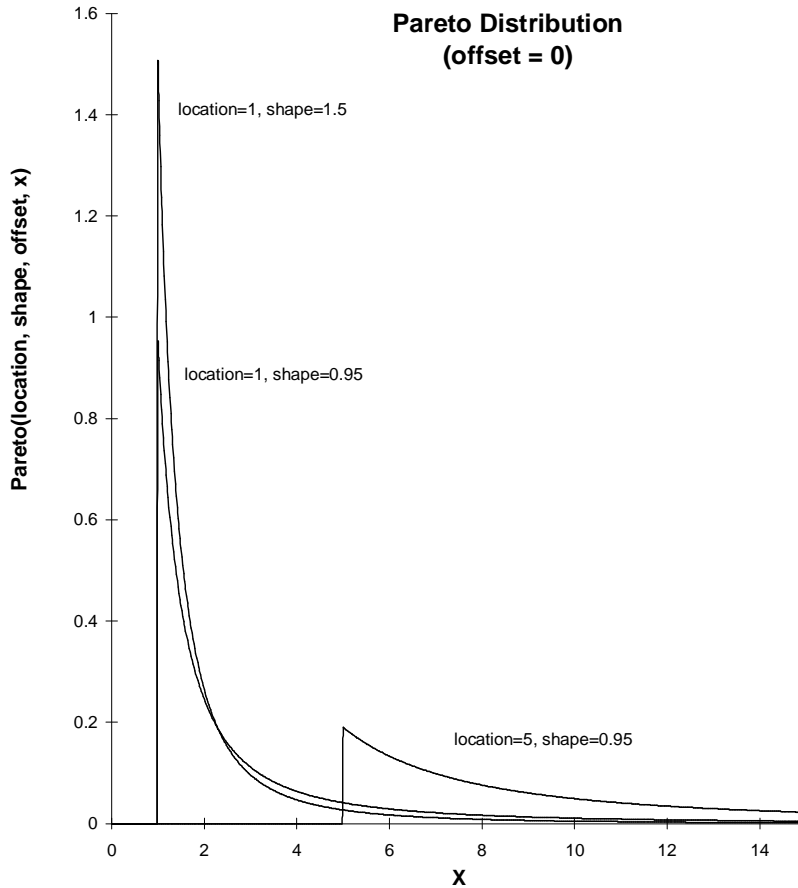


The Normal distribution is often used to pick message sizes. For instance, a message could be described as having a mean size of 20,000 bytes and a standard deviation of 5,000 bytes.

Normal (mean, standard deviation, stream)

## 12. Statistical Distribution Functions

### 12.10 The Pareto Distribution



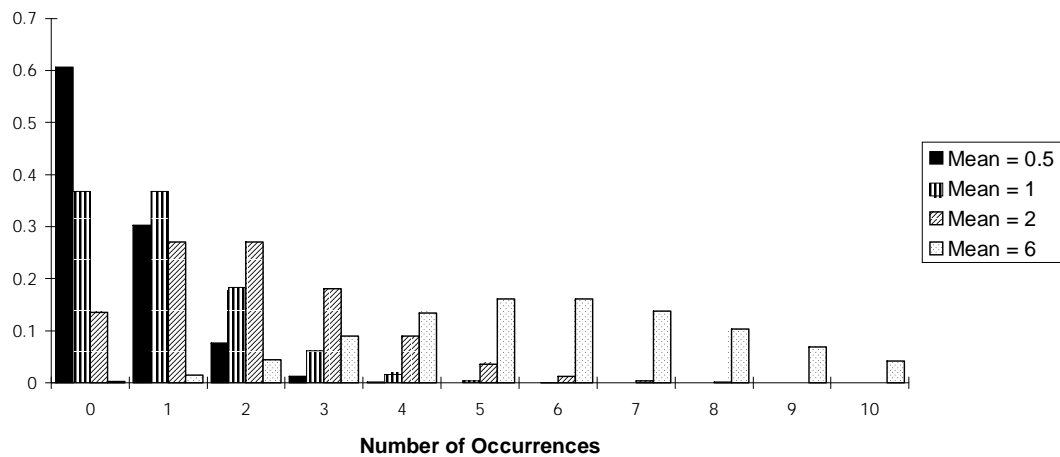
Pareto(location, shape, offset, stream)

The Pareto Distribution is a simple power-law type distribution that is useful for modeling bursty sources. The distribution is heavily peaked but the tail falls off relatively slowly. It takes three parameters: location, shape, and offset. The location specifies where the distribution starts, the shape specifies how quickly the tail falls off, and the offset shifts the distribution. Because of the heavy tail, the Pareto distribution can have infinite mean and variance. If the shape parameter is greater than 2, both the mean and variance are finite; if the shape parameter is greater than 1 but less than or equal to 2, the mean is finite but then variance is infinite, if the shape parameter is less than or equal to 1, both the mean and variance are infinite.

## 12.11 The Poisson Distribution

The Poisson distribution function takes one positive, real number, the mean. It returns a positive, integer. The stream parameter specifies which random number stream will be used to provide the sample. Note that this distribution, when provided with a time interval, returns an integer which is often used to represent the number of arrivals which are likely to occur in that time interval. Note that, in simulation, it is more useful to have this information expressed as the time interval between successive arrivals. For this purpose, use the Exponential distribution. Using the Exponential distribution to sample interarrival times will result in an arrival pattern characterized by the Poisson distribution.

**Poisson(Mean, Number)**

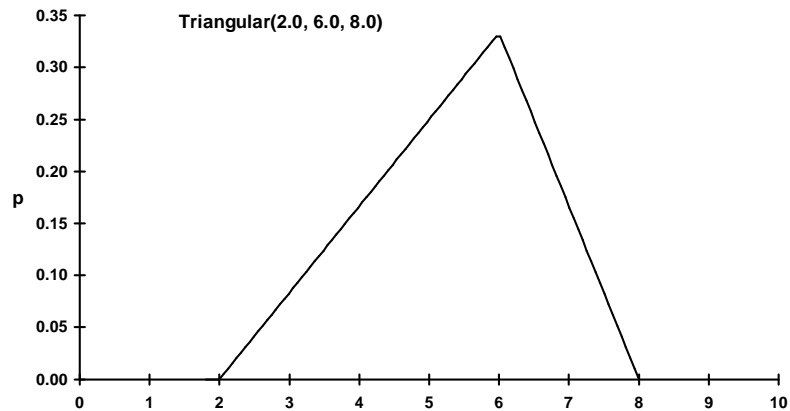


Poisson (mean, stream)

## 12. Statistical Distribution Functions

### 12.12 The Triangular Distribution

The Triangular distribution function takes three positive, real numbers: min, mode and max. It returns a positive, real number which is a sample from that range. The stream parameter specifies which random number stream will be used to provide the sample.



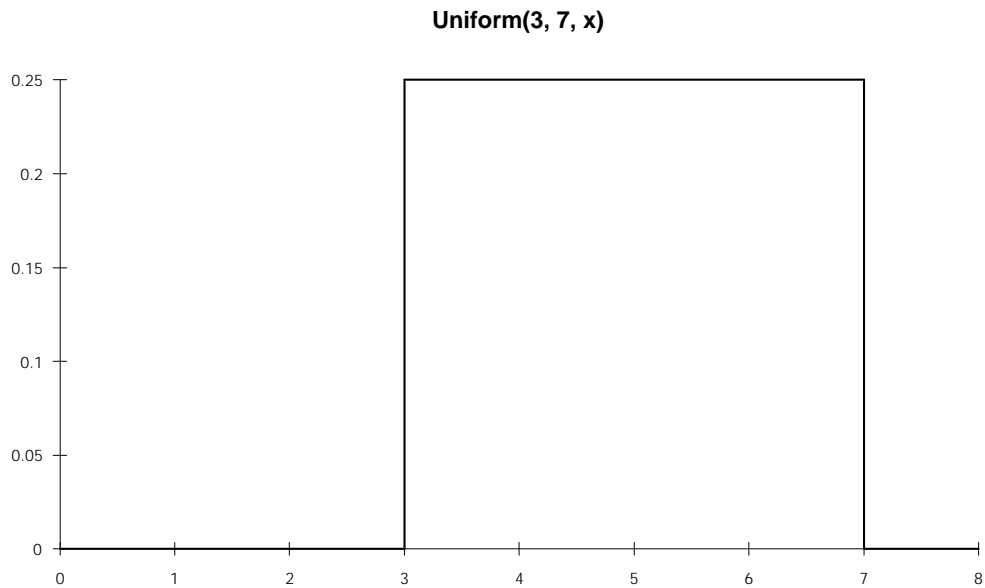
Triangular (min, mode, max, stream)

Note that the middle parameter is the mode, not the mean. However, if the triangle is symmetrical, then the mode value is also the mean value. This would be the case if the above example was changed to Triangular(2.0, 6.0, 10.0)



## 12.13 The Uniform Distribution

The Uniform distribution function takes two positive, real numbers: min and max. It returns a positive, real number. The stream parameter specifies which random number stream will be used to provide the sample. All values between min and max are equally probable, excluding min and max.



Uniform (min, max, stream)

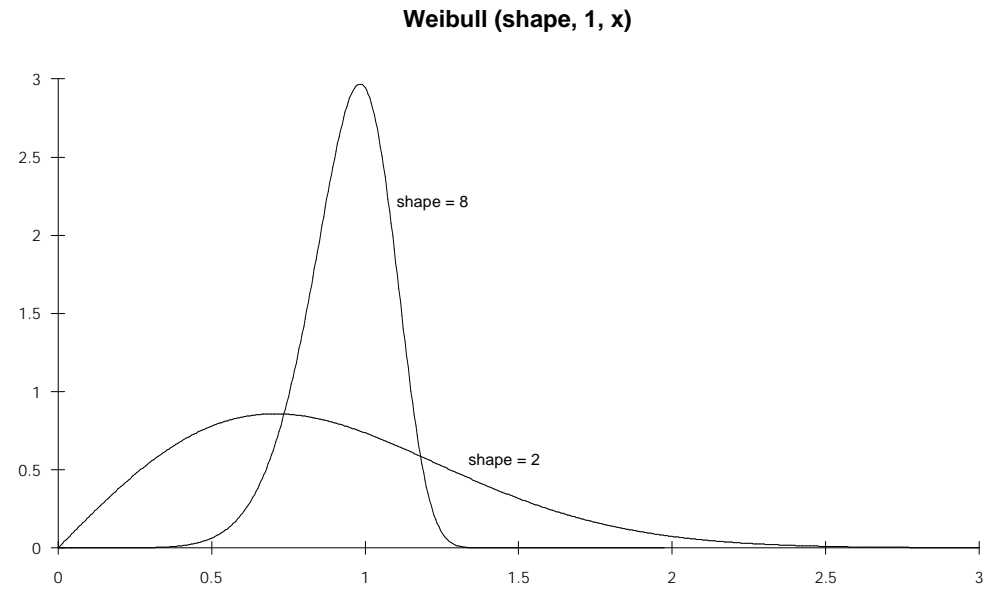
The integer version of this distribution function takes two integers: min and max. It returns an integer. All values between min and max are equally probable including min and max.

## 12. Statistical Distribution Functions

### 12.14 The Weibull Distribution

The Weibull distribution function takes two positive, real numbers: shape and scale. It returns a positive, real number. The stream parameter specifies which random number stream will be used to provide the sample.

A Weibull with shape and scale parameters of 1.0 is equivalent to an exponential distribution.



Weibull (shape, scale, stream)

## 12.15 User Distributions

User Distributions are a convenience feature provided by COMNET III. A User distribution is simply one of the standard probability distribution functions built in to COMNET to which the user has attached specific parameters.

Consider a large model in which the message sizes are generated using the same distribution with the same parameters through most of the model. Rather than typing in the parameters for the distribution each time, the user can define a user distribution, provide it with parameters and give it a name. Then this user-defined distribution can be specified anywhere it is needed. This also makes it clear that a particular distribution with a specific set of parameters is being used.

User distributions have another advantage. Consider the situation described above where a particular user distribution has been used to model message size throughout a model. By simply editing the user distribution, a global change to message sizes can be made throughout the model.

## 12.16 Table Distributions

COMNET III provides another user-defined distribution which is even more flexible and powerful. This is the table distribution. A table distribution allows you to use actual or empirical data to describe the distribution of a discrete or continuous random variable. Such data are typically in the form of histograms or frequency plots. To use a table distribution for a model parameter, you need only provide its name.

The custom distribution you describe can be either continuous or discrete. These are called respectively step or linear tables in COMNET. Step functions are often needed to model situations in which there are only a discrete number of choices. One example is a situation in which message sizes can only be certain values. For instance, 512 bytes with a 20% probability, 1,024 bytes with a 30% probability, and 2,048 bytes with a 50% probability.

You define table distributions by choosing the Define / Tables... menu selections. This presents the Table Detail dialog box which allows you to define the distribution. Selecting View gives you the opportunity to preview the distribution to ensure that it matches your expectation.

The TableType indicator tells how to interpret the values in the probability table. When step type is chosen, the described distribution will be discrete. Only those values explicitly specified by the user in the list can be returned by this table. When the linear type is set, the distribution will be continuous and COMNET will interpolate values between the ones provided by the user.

Each probability table contains a list of cumulative probabilities and their associated values. The probability values must be in ascending order, with the last entry in the list having a cumulative probability of one.

Once a table distribution has been defined, it is automatically added to the list of available probability distributions in COMNET's dialog boxes.

## 12. Statistical Distribution Functions

# 13. Menus

## 13.1 File Menu

File                                    Edit View Layout Create Define Simulate Report Library Help

New                                    Ctrl+N

Open                                    Ctrl+O

-----

Save                                    Ctrl+S

Save As...

-----

Import

External Model (\*.c3e)...

Network Topology...

Network Traffic

Event Trace

Messages...

Sessions...

Calls...

Bitmap

Export

Ascii Model Files (\*.c3)...

External Model (\*.c3e)...

Simulation Statistics...

EPS Postscript...

Raster Bitmap

Merge...

-----

Print...                                Ctrl+P

Exit



### 13.3 View Menu

File Edit View                      Layout Create Define Simulate Report Library Help

By List...

-----  
Zoom In

Zoom Out

Fit to Window

View (1:1)

-----  
Hide

Show

Show All

Toggle Names

-----  
Toolbars

Menubar

Colorbar

3D Bar

-----  
Enter

    Ctrl+E

Leave

    Ctrl+L

## 13. Menus

### 13.4 Layout Menu

File Edit View Layout

Create Define Simulate Report Library Help

Layout Size

-----  
Scale...

Move           Ctrl+M

Align...

Distribute...

-----  
Group

Ungroup

Bring to Front

Send to Back

-----  
Foreground Color

Background Color

Snapshot Measure Color

Reset Color

-----  
Snap

On

Off

Grid

On

Off

Spacing

Color



**13.5 Create Menu**File Edit View Layout CreateDefine Simulate Report Library HelpNodes

- Processing Node
- Router
- Computer Group
- Switch

Links

- Point-To-Point
- DAMA
- Aloha
- CSMA
- CSMA/CD
- Polling
- Token Passing
- CSMA/CA
- FDDI Basic
- Priority Token Ring
- Priority FDDI

Clouds

- WAN Clouds
- Frame Relay
- ATM

Application Sources

- Application Source

Message Sources

- Message Source
- Response Source

Session Sources

- Session Source

Call Sources

- Call Source

Cloud VCs

- Cloud VC

Cloud Access Link

- Cloud Access

## Remotes

- Remote Link
- Remote Node
- Socket

## Networks

- Network

## 13. Menus

### 13.6 Define Menu

File Edit View Layout Create Define Simulate Report Library Help

Node Parameters...  
Link Parameters...  
-----  
Global Commands...  
Global Variables...  
Protocols...  
Application Types...  
-----  
Backbone Properties...  
Routing Classes  
    Packets  
    Calls  
Routing Penalties  
    Packets  
    Calls  
-----  
User Distributions...  
Tabular Distributions...  
External Traffic  
-----  
Packet Rate Matrix...

## 13.7 Simulate Menu

File Edit View Layout Create Define Simulate

Report Library Help

Verify Model

Run Parameters

-----

Start Simulation F4

Halt Simulation F5

-----

Animate...

Trace...

## 13. Menus

### 13.8 Report Menu

File Edit View Layout Create Define Simulate Report

Library Help

Set File Name...

Select File Name...

Browse Reports...

-----

Select Snapshots...

Take Snapshot...

## 13.9 Library Menu

File Edit View Layout Create Define Simulate Report Library Help

Edit Contents...  
Software Objects  
Load  
Unload

## 13. Menus

### 13.10 Help Menu

File Edit View Layout Create Define Simulate Report Library Help

Index

Dump Memory Usage...

About COMNET III...

# Index

## A

Access Link.....23, 47, 50  
    Statistics .....405  
Access Link Statistics .....341  
access links .....318  
Access Point.....21, 22, 23, 32, 305, 307  
    Connectivity .....32  
    Creating.....32  
    Editing.....32  
    Execution .....32  
    Fields.....34  
    Purpose.....32  
    Reporting .....34  
Accuracy .....2, 3, 4  
Acme Bank Model .....7  
Add To .....414  
Aloha.....22  
Analytical Methods.....2  
Animation .....8, 23, 407  
    Control .....322  
    Speed.....322  
    Step Size .....322  
Answer Message Command ....24, 25, 53  
    Connectivity .....53  
    Creating.....53  
    Editing.....53  
    Execution .....54  
    Fields.....55  
        Message Size Calculation .....55  
        Message Size Units .....55  
        Message Text .....55  
        Name .....55  
        Packet Routing Class .....55  
        Packetizing Delay .....55  
        Priority .....55  
        Transport Protocol .....55  
    Purpose.....53  
    Reporting .....55  
Application  
    Delay .....345  
    Instance .....345  
    Pending List .....345  
    Scheduling .....3  
    Suspended .....72  
Application Command.....345  
Application Run Length.....341

Application Source .....24, 25  
Application Source Reports .....341  
Applications .....24  
    End User .....2, 3, 4, 5, 6  
Arc .....21, 23, 407, 409, 410  
Archive Library.....21  
Archive Menu .....325  
Areas .....410  
Automatic Parameter Iterator.....337  
Averaged Statistics .....405  
**B**  
Backbone .....22, 26  
    Model Layout.....305  
    User System .....3, 4, 5, 7  
Backbone Links .....32  
Bandwidth  
    Circuit Switched373, 374, 375, 376, 377,  
        378, 379  
Banking Network .....4  
batch mode .....8  
binary files .....30  
Bitmap.....306, 407  
Block  
    Call.....373, 376, 377  
    Packet.....360, 370, 371, 372  
Blocked Call Counts .....341  
Bottleneck .....3, 4, 6, 9  
bridges.....21  
Browse Reports.....343  
Buffer Processing.....35  
    Backward Explicit Congestion Notifica-  
        tion (BECN).....36  
    Buffer Policy.....35  
    Buffer Size .....35  
    Buffer Sorting .....36  
    Explicit Congestion Notification ...36  
    Forward Explicit Congestion Notification  
        (FECN).....36  
    Preemption Policy.....36  
    Processing at Buffers .....37  
    Threshold Policy .....36  
Building Blocks .....3  
burst size .....50  
bursty sources .....426  
Bursty Traffic.....4, 5

- Busy
  - Link .....352
  - Node .....344, 347
- Byte .....21
- C**
- Call Counts .....341
- Call Holding Time .....373, 374, 375
- Call Level.....341
- Call Source.....25, 373, 374, 375
  - Call Holding Time distribution373, 374, 375
  - Destination ...373, 374, 375, 376, 377
- Call Source Reports .....341
- Canvas .....22, 23, 409
  - Dimension .....413
- Canvas Window .....408, 410, 411, 412
- Carrier-Sense Multiple-Access .....22
- Cell-Relay Network .....318
- Cell-Relay Networks.....40
- Channel Utilization .....341
- Circle.....407, 409
- Circuit Switched Network .....2
- Circuit Switched Traffic318, 373, 374, 375, 376, 377, 378, 379
- Circuit-Switched Model.....38
  - Connectivity .....38
  - Creating.....38
  - Editing.....38
  - Execution .....38
  - Fields.....39
  - Purpose.....38
  - Reporting .....39
- Clear
  - Menu Option .....412
  - Pending Messages .....384, 385
- client-server operations .....73
- Cloud.....21, 22, 23, 40, 318
  - Behavior .....40
  - Congestion Levels.....40
  - Congestion States.....42
  - Connectivity .....41
  - Creating.....41
  - Editing.....42
  - Fields.....43
    - Delay Switch.....44
  - Frame Assembly .....44
  - Frame Drop Probability .....45
  - Frame Max .....44
  - Frame Min.....44
  - Frame Overhead.....44
  - Frame Priority .....44
  - Parameter Set Name.....44
  - Routing Interval .....44
  - Session Limit .....44
  - Transmit Non-Virtual Circuits (VCs) 44
  - Use of Comitted Information Rate (CIR) in Virtual Circuits (VC) .....44
  - Handling Frame Dropping .....40
  - Purpose.....40
- Cloud Access Link.....47
  - Connectivity.....47
  - Creating.....47
  - Editing.....47
  - Fields.....47
    - Buffer Definition.....48
    - Buffer Limit .....48
    - Buffer Policy.....48
    - Buffer Units .....48
    - Early Packet Discard/Partial Packet Discard .....48
      - Buffer Threshold (Buffer Units) 48
      - Discard Type.....48
    - Entry Bandwidth .....47
    - Exit (Egress) Buffer .....48
    - Exit Bandwidth .....47
    - Number of Circuits .....47
    - Propagation .....48
    - Use Unlimited buffer .....48
    - Purpose.....47
- Cloud Access Links .....319
- Cloud VCs.....319
- Cloud Virtual Circuit .....50
  - Connectivity .....50
  - Creating.....50
  - Editing.....51
  - Fields.....51
    - Burst Type.....51
    - Committed Burst Size .....51



Committed Information Rate (CIR) or Burst Interval.....51	Tool Tips and Status Bar Messages327
Excess Burst Size.....51	Using the Tool Palette.....303
Mark Discard Eligible (DE) Frames 51	Computer & Communication Node.3, 14
Never Drop Due to Excess Burst52	Computer and Communications Node.21
Number of Switches.....52	Computer Group Node.....21
Propagation Delay.....52	Computers.....21
Show Burst Size.....52	Connection Oriented Routing2, 318, 349, 383
Purpose.....50	Connection Tool .....2, 14
Collision.....351	Connectionless Routing .2, 318, 349, 383
Collision Statistics .....341	Connection-Oriented Routing.....26
Color .....409, 411	Contention Interval .....352
Palette.....408	Continuous Probability Distributions 431
Command	Copy
Processing .....320, 344	From Clipboard.....310
Read .....344, 347	Simdraw .....412
Response .....358, 371	To Clipboard.....310
Session .....359, 372, 381, 384	Cost
Transport.....358, 371	Dollar .....1, 6
Write .....344, 347	Count.....406
command line interface.....8	Create
command repertoire .....24	Application.....318
committed burst size .....50	Call.....318
committed information rate .....50	File & Subdirectory .....7, 308
COMNET Baseline.....80, 415	Group In Simdraw .....414
Network Load Characterization...416	Icon .....306, 407
Network Topology Information...416	Link.....14, 318
Overview.....415	Menu .....318
System Architecture.....415	Message .....318
COMNET III	Network Description.....1, 308
3D View .....327	Node.....14, 318
Arc Editing.....327	Packet.....344, 360, 370, 371, 372
Background Text.....328	Pending Message Notice.....384, 385
Bitmap Import.....327	Session .....318, 380, 381
Color Palette .....327	Session Connect Packet372, 380, 381
Dockable Palettes and Toolbar ....327	Session Setup Packet372, 380, 381, 384, 385
Export Encapsulated PostScript...327	Tag in Simdraw.....414
Export Raster Bitmap.....328	Create Menu.....318
Model Editing Features.....328	CSMA .....22, 351
Object Creation Tools.....304	Contention Interval .....352
Report Request Manager .....328	CSMA/CA Link.....22
Shapes .....327	CSMA/CD .....22
starting .....7	Cut
Starting the Program .....303	COMNET III.....305
Terminology.....21	Simdraw .....412

- Cycle
  - Simdraw .....409
- Cycle Time.....5, 320
- D**
- data traffic .....26
- DE .....50
- DECNet.....27
- Define Menu .....320
- Delay
  - Queuing.....5
- Demand Assigned Multiple Access Link22
- Demo Model .....7
- Desintaion
  - Random Neighbor.....86
- Destination .....4, 346, 356
  - Call Source...373, 374, 375, 376, 377
  - Least Busy List .....81
    - Connectivity .....81
    - Creating.....81
    - Editing.....81
    - Execution .....81
    - Fields.....81
      - Destination Node Name....81
    - Purpose.....81
    - Reporting .....81
  - Message357, 358, 359, 360, 370, 371, 372
  - Multicast List .....82
  - Random List.....84
  - Random Neighbor.....86
  - Round Robin List.....87
    - Connectivity .....87
    - Creating.....87
    - Editing.....87
    - Execution .....87
    - Fields.....87
      - Destination Node Name....87
    - Purpose.....87
    - Reporting .....87
  - Session .380, 381, 382, 383, 384, 385
  - Weighted List.....88
- Destination List356, 357, 358, 359, 360, 370, 371, 372, 373, 374, 375, 380, 381
- Dimension
  - Canvas.....413
  - Icon .....413
- Disconnected CallCounts.....341
- Discrete Event Simulation .....2, 4, 322
- Discrete Probability Distributions .....431
- Disk Access Error Counts.....341
- Disk Access Time .....5
- Disk Capacity.....344, 347
- Distribution
  - Beta .....417
  - Erlang.....418
  - Exponential .....419
  - Gamma.....420
  - Geometric.....421
  - Hyperexponential.....422
  - Integer .....423
  - Lognormal.....424
  - Normal .....425
  - Poisson .....427
  - System.....90
    - Connectivity .....90
    - Creating.....90
    - Editing.....90
    - Execution .....90
    - Fields.....91
      - Beta .....91
      - Erlang.....91
      - Exponential .....91
      - Gamma.....91
      - Geometric.....91
      - Hyperexponential.....91
      - Integer .....91
      - Lognormal.....91
      - Normal .....91
      - Poisson .....91
      - Triangular.....91
      - Uniform.....91
      - Weibull.....91
    - Purpose.....90
      - Random Number Generation Al-  
gorithm.....90
      - Random Number Starting Seed  
90
    - Reporting .....90
- Table .....92
  - Connectivity .....92
  - Creating.....92

Editing.....	92	Erlang Distribution .....	418
Execution .....	92	Ethernet.....	3, 5, 22
Fields.....	92	excess burst size.....	50
Name.....	92	Exit Buffer Size .....	405
Probability.....	93	Exit Link Utilization .....	405
Table Type .....	93	Exponential Distribution.....	419, 427
Value.....	93	Export.....	308
Purpose.....	92	Exporting Statistics .....	406
Reporting .....	92	Extended Mode .....	14, 304, 306, 409
Triangular.....	428	external file format.....	30
Uniform.....	429	External Model File .....	30
UniformInteger .....	429	Export	
User Defined.....	94	Defaults.....	30
Connectivity.....	94	TRUE_NAME .....	30
Creating.....	94	External Sources .....	25
Editing.....	94	External Traffic.....	321
Execution .....	94	External Traffic File.....	25
Fields.....	94	External Traffic Sources .....	25
Name.....	94		
Probability Distribution .....	94	<b>F</b>	
Purpose.....	94	FDDI.....	22
Reporting .....	94	FDDI Link.....	22
Weibull.....	430	File	
Distribution Table .....	29	COMNET III Data File.....	7, 16
distribution, power-law .....	426	COMNET III Report File .....	7
distributions .....	29	File Size .....	3
Duplicate		File Transfer.....	3
Simdraw .....	412	file format .....	30
dynamic algorithms.....	26	File Menu.....	8, 308
<b>E</b>		Fill Style.....	408, 409, 411
Edit		Filter Message Command .....	24
Commands .....	320	Fields	
Graphics .....	306	Filter Command Name.....	74
Menu Option.....	310	Flow Control .....	97
Packet Routing Class .....	321	Connectivity.....	97
Packet Routing Penalties .....	321	Creating.....	97
Simdraw Editor .....	407	Editing.....	97
Table Distributions .....	321	Execution .....	97
Transport Layer Protocol .....	320	Fields.....	98
User Distributions .....	321	ACK Priority.....	98
Edit Features .....	328	End-To-End ACK Bytes .....	98
Edit Menu .....	305, 306	Retransmit Blocked Packets ...	98
End System .....	3, 5	Retransmit Time .....	98
Enter.....	313	Window Size.....	98
Entry Link Utilization.....	405	Fields for Enhanced Flow Control.	98
		Error Control Options .....	98

Holding ACKs for More Data	98	Reporting	104
Retransmit Timeout Backoff	99	TCP/IP Window	105
Retransmit Timeout Calculated from Estimated Round-Trip Time	99	Connectivity	105
Fixed Window	100	Creating	105
Connectivity	100	Editing	105
Creating	100	Execution	105
Editing	100	Fields	105
Execution	100	Purpose	105
Fields	100	Reporting	105
Purpose	100	Font Style	409
Reporting	100	Frame	
Jumping Window	102	Collision	351
Connectivity	102	Error probability	349
Creating	102	Report	349
Editing	102	Transmission Time	350
Execution	102	Frame Counts	341
Fields	102	Frame Delay	341
Purpose	102	Frame Relay	47
Reporting	102	Networks	40
Leaky Bucket	103	frames	23
Connectivity	103	from	414
Creating	103	Full Duplex	349
Editing	103		
Execution	103	<b>G</b>	
Fields	103	Gamma Distribution	420
Purpose	103	Gateway	5
Reporting	103	gateway node	22
Purpose	97	gateways	21
Reporting	98	Geometric Distribution	421
Sliding Window	101	Grade Of Service	4, 6
Connectivity	101	Granularity	4
Creating	101	Graphic	
Editing	101	Editing Points	412
Execution	101	Editor	407
Fields	101	Icon	407
Purpose	101	Image	407, 409
Reporting	101	Importing	306
SNA Pacing	104	Library File	306, 407
Connectivity	104	Primitive	407
Creating	104	Priority	411
Editing	104	Graphical User Interface 1, 303, 307, 407	
Execution	104	Graphical user interface	2
Fields	104	Graphing	
Purpose	104	Real Time	9
		Grid	413
		Grouping Images	413

## H

Help Menu .....	326
HP NetMetrix .....	321
Hub .....	22
Hyperexponential Distribution .....	422

## I

### Icon

Background Tool .....	306
Creating .....	407
Dimension .....	413
Link .....	9
Moving .....	307, 314
Node .....	9
Program .....	7
Scaling .....	314
Subnetwork .....	306
Text Tool .....	306

ID Code .....	305
---------------	-----

### Idle

Link .....	352
Node .....	344

IEEE 802 .....	106
----------------	-----

Standards .....	106
-----------------	-----

Broadband Technical Advisory Group .....	106
--	-----

CSMA/CD Networks .....	106
------------------------	-----

Fiber Optic Technical Advisory Group .....	106
--	-----

Higher Layer Interface .....	106
------------------------------	-----

Integrated Voice and Data LAN Interface .....	106
---	-----

Logical Link Control .....	106
----------------------------	-----

Metropolitan Area Networks .....	106
----------------------------------	-----

Standard for Interoperable LAN Security .....	106
---	-----

Token Bus Networks .....	106
--------------------------	-----

Token Ring Networks .....	106
---------------------------	-----

Wireless LAN .....	106
--------------------	-----

IGRP .....	26
------------	----

Importing Network Traffic Files .....	308
---------------------------------------	-----

Importing NMS Topology Files .....	308
------------------------------------	-----

Input Buffer Totals .....	341
---------------------------	-----

Input Buffer Use .....	341
------------------------	-----

Installation of COMNET III .....	7
----------------------------------	---

Integer Distribution .....	423
----------------------------	-----

Interarrival time .....	419
-------------------------	-----

internal file format .....	30
----------------------------	----

Internetworking .....	4
-----------------------	---

IPX .....	27
-----------	----

ISDN .....	2, 22
------------	-------

Iteration Time .....	24
----------------------	----

## K

Kelton, W. David .....	3
------------------------	---

Kilobyte .....	21
----------------	----

## L

LAN .....	2, 3, 5, 7
-----------	------------

Law, Averill .....	3
--------------------	---

Layer 4 .....	320
---------------	-----

Leased Line .....	4, 5
-------------------	------

Least Busy List .....	81
-----------------------	----

Level Of Detail .....	4, 5
-----------------------	------

libraries .....	29
-----------------	----

### Library

Graphic Images .....	306, 407, 413
----------------------	---------------

Network Components .....	2, 318
--------------------------	--------

Line Style .....	409, 411
------------------	----------

Line Width .....	409, 411
------------------	----------

Lines .....	410
-------------	-----

Link .....	21, 22, 107
------------	-------------

Aloha .....	121
-------------	-----

Connectivity .....	121
--------------------	-----

Creating .....	121
----------------	-----

Editing .....	121
---------------	-----

Execution .....	122
-----------------	-----

Fields .....	124
--------------	-----

Control Node .....	124
--------------------	-----

Current State .....	124
---------------------	-----

Icon .....	124
------------	-----

Name .....	124
------------	-----

Statistics .....	124
------------------	-----

Time Of Next State Change .....	124
---------------------------------	-----

Time to Failure .....	124
-----------------------	-----

Time To Repair .....	124
----------------------	-----

Parameter Set Fields .....	124
----------------------------	-----

Bandwidth .....	124
-----------------	-----

Control Channel .....	125
-----------------------	-----

Framing Characteristics .....	125
-------------------------------	-----

Propagation Delay .....	124
-------------------------	-----

Retry Interval .....	125
----------------------	-----

Binary Exponential Backoff .....	
----------------------------------	--

- 125
  - Probability Distribution 125
  - Session Limit ..... 124
  - Slot Width ..... 124
  - Transmission Return Rate 125
- Purpose..... 121
- Reporting ..... 124
- Binary Exponential Backoff ..... 109
  - Connectivity ..... 109
  - Creating ..... 109
  - Editing ..... 109
  - Execution ..... 109
  - Fields ..... 109
    - Limit Delay ..... 110
    - Offset ..... 110
    - Retry Limit ..... 110
    - Slot Time ..... 110
    - Stream ..... 110
  - Purpose..... 109
  - Reporting ..... 109
- Connections ..... 305
- Connectivity ..... 107
- Creating ..... 107
- Creation ..... 14, 304
- CSMA ..... 126
  - Connectivity ..... 126
  - Creating ..... 126
  - Editing ..... 126
  - Execution ..... 127
  - Fields ..... 128
    - Control Node ..... 128
    - Current State ..... 129
    - Icon ..... 128
    - Name ..... 128
    - Statistics ..... 129
    - Time Of Next State Change 129
    - Time To Failure ..... 129
    - Time To Repair ..... 129
  - Parameter Set Fields ..... 129
    - Bandwidth ..... 129
    - Collision Window ..... 129
    - Control Channel ..... 130
    - Framing Characteristics ... 129
    - Interframe Gap ..... 129
    - Jam Interval ..... 129
    - Propagation Delay ..... 129
  - Retry Interval ..... 129
    - Binary Exponential Backoff
      - 130
        - Probability Distribution 130
        - Session Limit ..... 129
        - Transmission Return Rate 130
    - Purpose..... 126
    - Reporting ..... 128
  - CSMA/CD ..... 135
    - Connectivity ..... 136
    - Creating ..... 135
    - Editing ..... 136
    - Execution ..... 136
    - Fields ..... 137
      - Control Node ..... 137
      - Current State ..... 137
      - Icon ..... 137
      - Name ..... 137
      - Statistics ..... 137
      - Time Of Next State Change 137
      - Time To Failure ..... 137
      - Time To Repair ..... 137
    - Parameter Set Fields ..... 137
      - Bandwidth ..... 137
      - Binary Exponential Backoff 138
      - Collision Window ..... 137
      - Control Channel ..... 138
      - Framing Characteristics ... 138
      - Interframe Gap ..... 138
      - Jam Interval ..... 137
      - Probability Distribution ... 138
      - Propagation Delay ..... 138
      - Retry Interval ..... 138
      - Session Limit ..... 138
      - Transmission Return Rate 138
    - Purpose..... 135
    - Reporting ..... 137
  - Editing ..... 107
  - Failure ..... 5, 9
  - Fields ..... 107
    - Connections ..... 108
    - Control Node ..... 107
    - Current State ..... 108
    - Icon ..... 107
    - Name ..... 107
    - Parameters ..... 107

Time Of Next State Change ...	108	Session Limit .....	146
Time To Failure .....	107	Purpose.....	143
Time To Repair .....	107	Reporting .....	145
Type .....	107	Polling.....	148
Framing.....	111	Connectivity.....	148
Connectivity.....	111	Editing.....	148
Creating.....	111	Execution .....	148
Editing.....	111	Fields.....	150
Execution .....	111	Control Node.....	150
Fields.....	112	Icon .....	150
Automatically Retransmit Frame		Link State .....	150
Errors .....	112	Name.....	150
Frame Error Probability ...	112	Statistics .....	150
Frame Max.....	112	Parameter Set Fields .....	150
Frame Min.....	112	Bandwidth.....	150
Frame Assembly .....	111	Control Channel.....	150
Frame Overhead.....	112	Framing Characteristics ...	151
Purpose.....	111	NAK (Negative Acknowledgment)	
Reporting .....	112	Delay .....	150
ISDN and SONET Parameter Sets	113	Poll Delay .....	150
Point To Point Protocol.....	4	Poll Needed To Send .....	150
Point-To-Point .....	143	Propagation Delay.....	151
Connectivity.....	143	Session Limit .....	151
Creating.....	143	Transmission Return Rate	150
Editing.....	143	Purpose.....	148
Execution .....	144	Reporting .....	149
Circuit Switching .....	145	Port.....	152
Packet Switching.....	144	Connectivity.....	152
Fields.....	145	Creating.....	152
Control Node.....	146	Editing.....	152
Icon .....	146	Execution .....	152
Link State .....	146	Fields.....	153
Name.....	145	Call Routing Penalty Table	155
Statistics .....	146	Input Buffer Delay .....	153
Parameter Set Fields		Input Buffer Size.....	153
Circuit Switching .....	146	Line Card ID .....	154
Bandwidth.....	146	Output Buffer Delay .....	154
Bandwidth Reserved for 1-		Output Buffer Size .....	154
Hop Calls .....	146	Packet Routing Penalty Table	155
Number Of Circuits ...	146	Purpose.....	152
Packet Switching.....	146	Reporting .....	152
Bandwidth From Node 1	146	Purpose.....	107
Bandwidth From Node 2	146	Token Passing .....	157
Framing Characteristics	147	Connectivity.....	157
Number Of Circuits ...	146	Creating.....	157
Propagation Delay.....	146	Editing.....	157

Execution .....	157	Menu Option .....	307
Fields .....	158	Create	
Icon .....	159	Application Source .....	318
Link State .....	159	Call Sources .....	318
Name .....	158	Cloud Access Links .....	319
Statistics .....	159	Cloud Virtual Circuits.....	319
Parameter Set Fields .....	159	Clouds .....	318
Bandwidth .....	159	Links .....	318
Framing Characteristics ...	159	Message Sources .....	318
Propagation Delay .....	159	Networks .....	319
Session Limit .....	159	Nodes .....	318
Token Holding Limit .....	159	Remotes .....	319
Token Passing Delay .....	159	Session Sources.....	318
Purpose.....	156, 157	Define	
Reporting .....	158	Application Types.....	320
Utilization .....	16	Backbone Routing.....	320
Link Reports .....	341	External Traffic.....	321
Link-State Shortest Path First .....	26	Global Commands .....	320
Load .....	1, 3, 4, 5, 413	Global Variables .....	320
Loading A Model.....	8	Link Parameters .....	320
Loading COMNET III .....	8	Node Parameters .....	320
Loading Graphic Libraries .....	413	Packet Rate Matrix.....	321
Lognormal Distribution .....	424	Packet Routing Penalties .....	321
<b>M</b>		Routing Class .....	321
Main Menu.....	307	Tables.....	321
Mainframe.....	1, 3, 4	Transport Protocols.....	320
Mark Style.....	409	User Dists.....	321
Maximum.....	406	Define Backbone Routing.....	15
Mean Entry Link Utilization.....	405	Edit	
Mean Exit Buffer Size .....	405	Clear.....	310
Mean Exit Link Utilization .....	405	Clone .....	310
Megabyte .....	21	Copy .....	310
Menu .....	433	Cut.....	310
Archive.....	325	Duplicate .....	310
Create .....	318, 437	Find .....	310
Define.....	320, 438	Parent .....	310
Edit.....	305, 306, 310, 434	Paste .....	310
File .....	8, 308, 433	Properties .....	310
Help.....	326, 442	Select All.....	310
Layout .....	436	Select by List .....	310
Library .....	441	Edit Detail .....	305, 306
Main .....	307	Edit Move .....	307
Report.....	440	File	
Simulate .....	8, 322, 439	Exit.....	309
View .....	311, 435	Export.....	308
		Import.....	308



Merge .....	309	Set File Name.....	324
New .....	308	Setup Commands .....	324
Open.....	308	Take Snapshot.....	324
Print.....	309	Transport Commands.....	324
Save.....	308	WAN Clouds.....	324
Save As .....	308	Reports/Call Sources.....	16
File Export .....	308	Simulate	
File Import .....	308	Animate.....	322
File Save As .....	16	Halt.....	322
Help		Run Parameters .....	322
About COMNET III.....	326	Start.....	322
Dump Memory Message.....	326	Trace .....	322
Index .....	326	Verify .....	322
ReadMe File.....	326	Simulate Run Parameters.....	15
Reference Manual .....	326	Simulate/Browse Reports .....	16
Layout		Simulate/Start.....	16
Align .....	314	Simulate/Trace .....	16
Background Color.....	317	Table Detail.....	431
Bring to Front.....	317	View	
Distribute .....	316	By List.....	311
Foreground Color.....	317	Enter.....	313
Grid .....	317	Fit to Window .....	312
Group .....	317	Hide.....	312
Layout Size .....	314	Leave.....	313
Move .....	314	Show .....	313
Reset Color .....	317	Show All .....	313
Scale.....	314	Toggle Names .....	313
Send to Back .....	317	Toolbars .....	313
Snap .....	317	View (1 to 1).....	312
Ungroup .....	317	Zoom In.....	312
Library		Zoom Out.....	312
Edit Contents.....	325	Merge .....	309
Software Objects.....	325	Message	
Report		Destination.....	357, 358, 359, 360, 370, 371, 372
Application Sources .....	324	Source .....	318
Background Packet Flows .....	324	Message Delay .....	341, 342
Browse Reports.....	324	Message ID .....	25
Call Sources .....	324	Message Response Source Reports.....	341
Global Setup Commands .....	324	Message Source .....	25
Global Transport and Answer Com- mands .....	324	Interarrival Time .....	160
Links .....	324	Applicability .....	160
Message and Response Sources.....	324	Creating.....	160
Nodes .....	324	Editing.....	161
Select Snapshots .....	324	Execution .....	161
Session Sources.....	324	Fields.....	162

Interarrival Time .....	162
Start Time .....	162
Stop Time .....	162
Purpose .....	160
Reporting .....	162
Message Text .....	170
Connectivity .....	170
Creating .....	170
Editing .....	170
Execution .....	170
Fields .....	171
Message Text .....	171
Copy Message Name .....	171
Set Message Text .....	171
Use Original Message .....	171
Purpose .....	170
Reporting .....	171
Packetizing Delay .....	163
Connectivity .....	163
Creating .....	163
Editing .....	163
Execution .....	163
Fields .....	164
Packetizing Delay .....	164
Purpose .....	163
Reporting .....	164
Priority .....	165
Connectivity .....	165
Creating .....	165
Editing .....	165
Execution .....	165
Fields .....	166
Priority .....	166
Purpose .....	165
Reporting .....	166
Size Calculation .....	167
Connectivity .....	167
Creating .....	167
Editing .....	167
Execution .....	167
Fields .....	167
Message Size .....	167
Based On File Size .....	168
Based On Message Size .....	168
Probability Distribution .....	167
Purpose .....	167
Reporting .....	167
Size Units .....	169
Connectivity .....	169
Creating .....	169
Editing .....	169
Fields .....	169
Size Unit .....	169
Purpose .....	169
Reporting .....	169
Message Switched Network .....	2
Message Traffic .....	4
Metropolitan Area Network .....	4
Microsoft Windows .....	2, 7
Minimum .....	406
Minimum Hop Routing .....	26
Minimum Penalty .....	26
Model	
Creation .....	303
Background Icon Tool .....	306
Connection Tool .....	305
Jointed Arc Tool .....	305
Right Angle Tool .....	305
Import Bitmaps .....	327
Link .....	304
Moving Around the Layout	
Panning .....	307
Moving Objects .....	307
Multiple Objects .....	304
Node .....	304
Object Creation Tools .....	304
Selecting a Group of Objects .....	306
Selection Tool .....	305
Subnetwork Tool .....	306
Text Tool .....	306
Traffic Source .....	304
Using the Tool Palette .....	303
Filename .....	303
Modify a Network Description .....	303
Model Building .....	1, 14, 21
Model Building .....	21
Model Editing Features .....	328
Find .....	328
Matching Rules .....	328
Options .....	328
Select by List .....	328
View by List .....	328

Filter "DEFAULT" Objects ...	328
Filter Items with Default Value	328
Model File .....	29, 95
External .....	95
Purpose.....	95
Modeling Constructs .....	31
Modeling Terminology .....	21
Moving Icons .....	307
Multicast List .....	82
Connectivity .....	82
Creating.....	82
Editing.....	82
Execution .....	82
Dropped Packets .....	83
Flow Control .....	83
Routing Alogorithm .....	83
Fields.....	83
Destination Node Name .....	83
Purpose.....	82
Reporting .....	83

## N

Named Distributions .....	29
Network	
Access Point.....	305, 307
ACME Bank Model .....	7
Analyzers .....	25
Backbone .....	3, 4, 305
Banking.....	4
Bearer .....	3
Capacity .....	4, 5
Description.....	303
Design .....	1, 4, 5
General.....	1
Grade Of Service .....	6
Internetworking.....	4
LAN .....	2, 3, 5
Load .....	3, 6
MAN .....	4
Operation .....	320
Peak Loading .....	5
Performance .....	2, 3, 6
Protocols .....	3
Resilience.....	5
Subnetwork .....	306
Topology.....	2, 3
User.....	6
Voice.....	4
WAN.....	2, 3
Network General Sniffer.....	321
Network Topology .....	21
Network Traffic and Workload.....	23
Node.....	21, 172
Computer Group .....	194
Connectivity.....	194
Creating.....	194
Editing.....	194
Execution .....	195
Fields.....	196
Call Bandwidth .....	196
Quantity .....	196
Source/Sink Only .....	196
Purpose.....	194
Reporting .....	196
Connectivity.....	172
Creating.....	172
Creation.....	304
Cycle Time.....	5, 320
Disk Access Time .....	5
Disk Capacity.....	344, 347
Editing.....	172
Failure .....	5, 9
Fields.....	172
Icon .....	172
Name .....	172
Parameters.....	172
Commands .....	173
Current State .....	173
Time of Next State Change	172
Time To Failure .....	172
Time To Repair.....	172
Type .....	172
Input Buffer.....	349
Output Buffer.....	349
Parameters.....	24
Processing Node .....	179
Connectivity.....	180
Creating.....	179
Editing.....	180
Execution .....	180
Buffer Processing.....	183
Circuit Switching .....	184

- Command Execution.....184
- No Time Slice .....181
- Processor Scheduling .....180
- With Time Slicing.....182
- Fields.....185
  - Call Routing Table.....185
  - Icon .....185
  - Name .....185
  - Node State.....185
  - Packet Routing Table.....185
- Parameter Set Fields .....185
  - App Ready The Longest ..186
  - Apps Have Priority .....187
  - Between Applications .....186
  - Between Apps And Packets186
  - Between Pkt Buffers .....187
  - Call Limit.....191
  - Command List.....192
  - Disk Sector Size.....192
  - Disk Size .....191
  - Disk Transfer Overhead...192
  - Disk Transfer Time .....192
  - Earliest Arrival.....187
  - Edit Processor .....185
  - File List.....192
  - File Name.....192
  - File Read Only Flag .....192
  - File Size .....192
  - First App Available.....186
  - Input Buffer Size.....190
  - Number Of Processors .....185
  - Output Buffer Size .....190
  - Packet Priority.....187
  - Packet Processing Uses Proces-  
sor.....188
  - Pkts Have Priority .....186
  - Port Processing Times ....191
  - Processing Time/Cycle ....188
  - Processing Time/KByte ...190
  - Processing Time/Packet...189
  - Processing Time/Setup Packet  
190
  - Processor Usage .....188
  - Read/Write Commands Use Proc-  
essor .....188
  - Ready The Longest .....187
  - Selection Rules .....186
  - Session Limit .....191
  - Source/Sink Only.....185
  - Time Slice .....189
  - Purpose.....179
  - Reporting .....184
  - Purpose.....172
  - Router.....197
    - Connectivity.....197
    - Creating.....197
    - Editing.....198
    - Execution .....200
      - Circuit Switching .....201
      - Packet Switching.....200
    - Fields.....202
      - Call Routing Table.....202
      - Icon .....202
      - Name .....202
      - Node State.....202
      - Packet Routing Table.....202
    - Parameter Set Fields .....202
      - Bus Count .....204
      - Bus Rate .....204
      - Input Buffer Limit.....204
      - Output Buffer Limit .....205
      - Port Processing Times ....204
      - Processing Time/Cycle ....202
      - Processing Time/KByte ...203
      - Processing Time/Packet...202
      - Processing Time/Setup Packet  
204
      - Time Slice .....202
    - Purpose.....197
    - Reporting .....202
    - Updated Router Library .....206
  - Switch .....174
    - Connectivity.....174
    - Creating.....174
    - Editing.....175
    - Execution .....175
      - Circuit Switching .....176
      - Packet Switching.....175
    - Fields.....177
      - Call Routing Table.....177
      - Icon .....177
      - Name .....177

Node State .....	177	Window .....	408, 409, 410
Packet Routing Table .....	177	Parameter Iterator .....	337
Parameter Set Fields .....	177	Parameter Sets .....	209
Input Buffer Max .....	178	Connectivity .....	210
Output Buffer Max .....	178	Creating .....	210
Parameter Set Name .....	177	Editing .....	210
Virtual Cut-Through .....	177	Execution .....	210
Purpose .....	174	Purpose .....	209
Reporting .....	177	Reporting .....	210
Types of .....	21	parameter sets .....	29
Node Reports .....	341	Parent .....	310
Nodes .....	21	PC .....	4
Normal Distribution .....	425	Peak Loading .....	4, 5
<b>O</b>		Penalty Tables .....	26, 211
Object Creation .....	304	Connectivity .....	211
Object Library Access .....	29	Creating .....	211
Object Library .....	29	Editing .....	211
Objects .....	1	Execution .....	214
Origin .....	4	Fields .....	214
Output Buffer Totals .....	341	Name .....	214
Output Buffer Use .....	341	Penalty Per Routing Class .....	214
Overview of Modeling Constructs .....	21	Threshold .....	214
Owner Name .....	406	Purpose .....	211
Owner Type .....	406	Reporting .....	214
<b>P</b>		penalty tables .....	23
Packet		Percentiles .....	403
Animation .....	8	Percentiles and Plots .....	403
Routing .....	321	Access Link Statistics .....	405
Switch .....	3	Exporting Statistics .....	406
Packet Delay .....	341, 342	statfile.xpt .....	406
Packet Rate Matrix .....	288	Count .....	406
Packet Switched Network .....	2	Maximum .....	406
Packet Switched Networks .....	207	Minimum .....	406
Connectivity .....	207	Owner Name .....	406
Creating .....	207	Owner Type .....	406
Editing .....	207	Replication .....	406
Execution .....	207	Statistics Type .....	406
Fields .....	208	Sum .....	406
Purpose .....	207	Sum of Squares .....	406
Reporting .....	208	Overview .....	403
packets .....	23	Post-processed Plots .....	403
Pad .....	3, 4, 21	Real-time Plots .....	404, 405
Painting .....	411	Smoothed Data .....	403
Palette .....	407	Statistics Request Buttons .....	403
Tool .....	2, 8, 303, 305	Virtual Circuit Statistics .....	406
		Burst Size .....	406

- Frame Delay .....406
  - Performance .....1, 2, 3, 5
  - Plots .....403
  - Point To Point Link.....4
  - Point-To-Point Link .....22
  - Poisson Distribution.....427
  - Polling .....22
  - Polygon .....407, 410
  - Polyline .....410
  - Port .....23
  - port parameters .....23
  - Portability.....2
  - power-law type distribution .....426
  - Preempted Call Counts .....342
  - Preemption .....26, 375, 376, 377
  - Print Server .....3
  - print server .....3
  - Priority
    - Call.....374, 375
    - Graphic.....411
  - Priority Token Ring Link.....22
  - private data networks .....318
  - Probability
    - Blocking.....376, 377
    - Distribution .....320, 417
  - Process & Disk Utilization Reports ...342
  - Process Data.....56
  - Process Data Command .....56
    - Connectivity .....56
    - Creating.....56
      - Global Command .....56
      - Library Command.....56
      - Local Command.....56
    - Editing.....56
    - Execution .....57
      - No Time Slicing.....57
      - With Time Slicing.....57
    - Fields.....57
      - Name .....57
      - Number Of Cycles .....57
        - Based On File Size.....58
        - Probability Distribution ....57
      - Number Of Cycles
        - Based On Message Size .....58
      - Purpose.....56
      - Reporting .....57
- Processing
  - Command.....320
  - processing time .....24
- Processor
  - Speed.....5
- Processor Command .....320
  - Number Of Cycles .....320
- Processor Disk Utilization .....341
- Propagation Delay.....50, 349, 350
- Protocol.....296
  - Call.....233
    - Connectivity .....233
    - Creating.....233
    - Editing.....234
    - Minimum Hop Routing.....234
    - Minimum Penalty Routing.....236
    - Purpose.....233
    - User Defined Routing Tables 238
  - Connectivity .....296
  - Creating.....296
  - Editing.....296
  - Execution .....297
  - Fields.....297
    - Ack Priority.....299
    - End-to-End ACK Bytes .....298
    - Error Control ACK .....299
    - Flow Control Method.....299
    - Name .....297
    - Negotiated Packet Window Sizes299
    - Packet Data Bytes .....297
    - Packet Overhead Bytes .....297
    - Padded Packets .....299
    - Retransmit Blocked Packets ..299
    - Retransmit Time .....298
    - Window Size.....298
- Max Idle Bandwidth .....252
- Minimum Delay .....252
- Minimum Hop Routing.....241
  - Deviation Percentage .....241
  - Preemption .....242
  - Routing Update Interval.....241
- Minimum Penalty Routing.....242
  - Deviation Percentage .....242
  - Preemption .....242
  - Routing Update Interval.....242
- Minimum Queue .....253

Minimum Session .....	253	Connectivity .....	86
Packet .....	244	Creating .....	86
IGRP Metric .....	247	Editing .....	86
Link State Shortest-Path Metric .....	248	Execution .....	86
Minimum Penalty Metric .....	248	Fields .....	86
OSPF .....	248	Purpose .....	86
Rip Minimum Hop Metric .....	248	Reporting .....	86
Shortest Delay Metric .....	249	Random Number Stream .....	322, 417
User-Defined Routing Tables .....	249	Random Variables .....	417
Purpose .....	296	Random Variance .....	2
Random List .....	254	Read .....	24
Reporting .....	297	Read Command .....	344, 347
Round Robin List .....	254	Read File Command .....	59
User Defined Routing Tables .....	242	Connectivity .....	59
Preemption .....	243	Creating .....	59
Primary Route Selection .....	242	Global Command .....	59
Secondary Route Selection .....	243	Library Command .....	59
Protocol Rate Control .....	216	Local Command .....	59
Available Rate .....		Editing .....	59
Purpose .....	217	Execution .....	60
Constant Rate .....	218	Busy The Processor .....	61
Purpose .....	218	Calculate Time Duration .....	60
Purpose .....	216	Lock The File .....	60
Network Congestion .....	216	Update File Information .....	61
Throttled Rate .....		Fields .....	61
Purpose .....	219	Bytes To Read Calculation .....	62
Variable Rate .....	220	Based On File Size .....	62
Purpose .....	220	Based On Message Size .....	62
Protocol Rate Controls .....		Entire File .....	62
Throttled Rate .....	219	Probability Distribution .....	62
Protocols .....	3, 4	File Modification Method .....	61
public data networks .....	318	Decrement File .....	61
<b>Q</b> .....		Delete File .....	61
Quick Start .....	7	Do Not Modify .....	61
<b>R</b> .....		File To Access .....	61
Random List .....	84	Name .....	61
Connectivity .....	84	Purpose .....	59
Creating .....	84	Reporting .....	61
Destination Node Name .....	85	Received Message Count .....	341, 346
Editing .....	84	Received Message Scheduling .....	221
Execution .....	85	Connectivity .....	221
Fields .....	85	Creating .....	221
Purpose .....	84	Editing .....	221
Reporting .....	85	Execution .....	221
Random Neighbor .....	86	Fields .....	222
		Delay .....	222

Message Text .....	222	File Warnings.....	347
Copy Message Name .....	223	Global Traffic Command.....	399
Set Message Text .....	223	Link Delays & Utilization.....	349
Use Original Message .....	223	Link Frame Size Report.....	353
Purpose.....	221	Link Reports .....	341
Reporting .....	222	Link Utilization by Application ...	354
received message scheduling		Link Utilization by Protocol .....	355
delay .....	222	Link Utilization Statistics for Calls	379
Received Message Text .....	24	Message & Response Source Reports	341
Recenter .....	414	Message Delays for Message & Response	
Remote .....	224	Sources.....	356
Remove From .....	414	Message Delays for Session Sources	357
Replication .....	406	Message Delays for Setup Commands	
Independent.....	3	359	
Length .....	3, 350, 378, 379	Message Delays for Transport & Answer	
replication reports .....	343	Commands .....	358
replication time .....	28	Message Delivery .....	361
Reports .....	2, 3, 7, 9, 225, 322, 341, 342	Multiple Report Windows .....	327
Alarms.....	401	Node Buffer Policy .....	348
Application Delays .....	345	Node Reports .....	341
Application Source Reports .....	341	Node Utilization.....	344
Blocked Call Statistics .....	373	Node Utilization Statistics for Calls	378
Browse Reports.....	343	Packet Delays for Session Sources	370
Buffer Input by Node .....	388	Packet Delays for Setup Commands	372
Buffer Input by Port .....	389	Packet Delays for Transport & Answer	
Buffer Output by Node .....	391	Commands .....	371
Buffer Output by Port .....	392	Packet Statistics for Message & Response	
Call Source Reports .....	341	Sources.....	360
Call Statistics by Link.....	377	Preempted Call Statistics .....	375
Call Statistics by Node.....	376	Purpose.....	225
Cloud Access Buffer Policy Report	397	Random Access Link Performance	351
Cloud Access Link.....	396	Received Message Count.....	346
Cloud Early/Partial Packet Discard Re-		Report Request Manager .....	328
port .....	398	Reporting .....	225
Cloud Throughput.....	394	Response and Answer Destinations	400
Cloud Virtual Circuit Frame Delay and		Selecting.....	342
Burst Size .....	395	Session Blocking by Session Command	
Connectivity .....	225	386	
Creating.....	225	Session Blocking by Session Source	387
Disconnected Call Statistics.....	374	Session Lengths by Session Source	385
Editing.....	225	Session Lengths by Setup Command	384
Execution .....	225	Session Setup by Link.....	383
Fields.....	225, 226	Session Setup by Node .....	382
Reports .....	225	Session Source Reports.....	342
Select All.....	226	Setup Command Reports .....	342
Show Group Node Detail.....	226	Setup Delays for Session Sources	380



Setup Delays for Setup Commands	381
Snapshot and Alarms	401
Transport & Answer Commands Reports	342
Transport Ack Delay	365
Transport Assembly Interval	366
Transport Burst Size	367
Transport Packet Flag Report	368
Transport Packet Size	369
Transport Retransmission	362
Transport Timeout	364
Transport Window and Packet Interval	363
WAN Cloud Reports	341
When Produced	342
Resilience	5
Response	
Time	4
Response Command	358, 371
Response Source	25
RIP Minimum Hop	26
Round Robin List	87
Router	3, 4, 5, 22, 197
Connectivity	197
Creating	197
Editing	198
Execution	200
Fields	202
Purpose	197
Reporting	202
Updated Library	206
Router Node	21
Routing Algorithm	2, 22, 26, 382, 383
Routing Class	26, 320
Call	227
Connectivity	227
Creating	227
Editing	227
Execution	228
Fields	228
Bandwidth Required	228
Call Retry Interval	228
Hop Limit	228
Name	228
Reroute Connections	229
Purpose	227
Reporting	228
Packet	230
Connectivity	230
Creating	230
Editing	230
Execution	231
Fields	231
Hop Limit	231
IGRP Routing Bandwidth Factor	232
IGRP Routing Delay Factor	232
Name	231
Reroute Connections	231
Session Retry Interval	231
Purpose	230
Reporting	231
Routing Protocol	
Call	233
Connectivity	233
Creating	233
Editing	234
Execution	234
Minimum Hop Routing	234
Minimum Penalty Routing	236
User Defined Routing Tables	238
Dynamic Alternate	239
First Available	240
Max Idle Bandwidth	240
Random List	240
Round Robin	240
Fields	241
Link Utilization Update Interval	241
Minimum Hop Routing	241
Deviation Percentage	241
Preemption	242
Routing Update Interval	241
Minimum Penalty Routing	242
Deviation Percentage	242
Preemption	242, 243
Primary Route Selection	242
Routing Update Interval	242
Secondary Route Selection	243
Purpose	233
Reporting	241
User Defined Routing Table	242

Packet.....	244	Scheduling .....	24
Connectivity .....	245	Scroll Bars.....	8, 307
Creating.....	244	Sector .....	407, 409, 410
Editing.....	245	Select	
Execution .....	245	Group Of Objects.....	306
IGRP Metric.....	247	Object.....	305
Bandwidth Factor.....	247	Selection Rules .....	186
Delay Factor.....	247	Selection Tool.....	14, 305
K1,K2,K3.....	247	Server.....	3, 5
Load .....	247	Service Provider.....	4, 6
Link-State Shortest-Path Metric		Session	
(OSPF) .....	248	Command.....	359, 372, 381, 384
Minimum Penalty Metric.....	248	Destination.....	380, 381, 382, 383, 384, 385
RIP Minimum Hop Metric.....	248	Source .....	318
Shortest Delay Metric .....	249	Session Length.....	342
User Defined Routing Tables.....	249	Session Level .....	341
First Available.....	251	Session Source Reports.....	342
Fields.....	254	Setup Command.....	25
Congestion Threshold Type.....	254	Setup Command Reports .....	342
Connection-Oriented Routing For		Setup Commands .....	342
Sessions.....	254	Setup Counts .....	342
Delay Update Interval .....	254	Setup Delay.....	342
Deviation Percentage .....	254	Setup Session Command .....	24, 63
Link Utilization Update Interval		Connectivity.....	63
254		Creating.....	63
Routing Update Interval.....	254	Global Command.....	63
Maximum Idle Bandwidth .....	252	Library Command.....	63
Minimum Delay .....	252	Local Command.....	63
Minimum Queue .....	253	Editing.....	63
Minimum Sessions.....	253	Execution .....	64
Purpose.....	244	Fields.....	66
Random List.....	254	Confirm Packet Size .....	66
Reports .....	254	Destination Type.....	67
Round Robin .....	254	Multicast List .....	67
Routing Table .....	15	Random List.....	67
Routing Tables		Random Neighbor.....	67
User Defined.....	15	Weighted List.....	67
routing tables.....	26	Message Interarrival Time .....	67
Run Length .....	3, 350, 378, 379	Message Size Calculation .....	67
Run Parameters .....	28	Message Size Units .....	67
<b>S</b>		Message Text .....	67
Sample Model.....	7	Messages Per Session .....	66
Satellite .....	3	Name .....	66
Save.....	7, 308, 322, 413	Packet Routing Class .....	66
Save To Library .....	413	Packetizing Time .....	67
		Priority .....	66

Setup Packet Size .....	66	Connectivity .....	260
Transport Protocol .....	66	Creating .....	260
Purpose .....	63	Editing .....	260
Reporting .....	66	Execution .....	260
Scheduling .....	65	Fields .....	260
Shortest Measured Delay .....	26	Export Stats After Run .....	261
SIMDRAW .....	407	Include percentiles in export .....	261
modes, setting .....	409	Number Of Replications .....	261
SIMGRAPHICS II .....	407	Replication Length .....	260
Graphics Editor .....	407	Reset System Every Replication	261
Managing the Library .....	413	Warmup Every Replication .....	261
SIMDRAW .....	407	Warmup Length .....	260
Canvas Dimensions and Coordi-		Purpose .....	260
nates .....	413	Reporting .....	260
Constructing Graphic Images .....	409	Tracing .....	262
Cutting and Copying Objects .....	412	Connectivity .....	262
Editing Points .....	412	Creating .....	262
Editor Windows .....	408	Editing .....	262
Exiting .....	414	Execution .....	262
Grouping Images .....	413	Fields .....	262
Importing Text .....	414	Destination .....	262
Moving, Painting, Resizing, and		Next On/Off Time .....	262
Changing Priority .....	411	Pause For Messages .....	263
Recentring Images .....	414	Single Step .....	263
Saving and Loading Objects .....	413	Purpose .....	262
Selecting Objects .....	410	Reporting .....	262
Setting Modes, Styles, Colors and		Simulation Control .....	28
Line Widths .....	409	Single Step .....	9, 322
The Library .....	413	Step Size .....	322
Using the Grid .....	413	Simulation Modeling & Analysis .....	3
Starting SIMDRAW .....	407	sliding interval .....	50
Simulate Menu .....	8, 28, 322	sliding window .....	27
Simulation		SNA .....	27
Animation .....	256	Snap .....	413
Connectivity .....	256	Socket .....	264
Creating .....	256	Assembly .....	265
Dynamic Color Change .....	259	Packetization .....	264
Editing .....	256	Purpose .....	264
Execution .....	256	Socket-Based Protocol Constraints .....	265
Fields .....	257	Sockets .....	27
On/Off .....	257	Software Modeling Features	
Step Size .....	257	Advanced .....	329
Time Of Next Change .....	257	SONET .....	22
Purpose .....	256	Spreadsheet .....	1, 4
Reporting .....	257	Start Simulation .....	28
Parameters .....	260		

Starting COMNET III .....	7
Stat1.rpt .....	343
statfile.xpt .....	406
Statistic Type .....	406
Statistical Distribution Functions.....	417
Beta Distribution .....	417
Erlang Distribution .....	418
Exponential Distribution.....	419
Gamma Distribution .....	420
Geometric Distribution .....	421
Hyperexponential Distribution .....	422
Integer Distribution.....	423
Lognormal Distribution .....	424
Normal Distribution .....	425
Pareto Distribution .....	426
Poisson Distribution.....	427
Table Distribution .....	431
Triangular Distribution .....	428
Uniform Distribution .....	429
User Distribution.....	431
Weibull Distribution .....	430
Statistical monitoring.....	342
Statistical Monitors .....	403
Statistical Reporting.....	28
Statistics	
Exporting .....	266
Link .....	267
Collect Statistics .....	267
Connectivity .....	267
Creating.....	267
Editing.....	267
Execution .....	267
Fields.....	267
Purpose.....	267
Reporting .....	267
Save Observations.....	267
View .....	267
Message .....	268
Connectivity.....	268
Creating.....	268
Editing.....	268
Execution .....	268
Fields.....	268
Collect Statistics .....	268
Initial X Axis Maximum..	269
Initial X Axis Minimum ..	269
Initial Y Axis Maximum..	269
Initial Y Axis Minimum ..	269
On/Off.....	269
Plot Parameters .....	268
Save Observations.....	268
Purpose.....	268
Reporting .....	268
Monitors.....	270
Buffer Monitors .....	271
Buffer Level.....	271
Burst Size .....	272
Call Bandwidth Level (Utiliza-	
tion).....	271
Frame Size .....	271
Inter Packet Interval.....	271
Inter-Assembly Interval ..	272
Message Delivery and Message	
Transmission Delays	271
New Link Monitors.....	271
New Message Destination Moni-	
tors .....	271
Packet Size .....	272
Packets in Retransmission Event	
271	
Retransmission Timeout ..	272
Retransmissions for Blocked	
Packets .....	272
Round-Trip Time (Ack Delay)	
272	
Window Size.....	271
Export to Excel .....	270
Monitor Statistics and Confidence In-	
tervals.....	270
Node Monitors .....	270
Call Bandwidth Level (Utiliza-	
tion).....	271
Processor Utilization.....	270
Storage .....	271
Option to Store Averaged Observa-	
tions.....	270
Plot Parameters .....	273
Connectivity.....	273
Creating.....	273
Editing.....	273
Execution .....	273
Fields.....	273

Averaging Interval .....	273
Histogram.....	273
Maximum.....	273
Minimum .....	273
Number of Bins.....	273
Number Of Points .....	273
Observed Data.....	273
Smoothed Data.....	273
Start Time .....	273
Stop Time.....	273
Purpose.....	273
Reporting .....	273
statistics.....	28
Statistics Button .....	28, 403
Step Size .....	322
Stream	
Random Number.....	322, 417
Subnet .....	22, 26
Subnets.....	21
Subnetwork .....	22, 274
Connectivity.....	274
Creating.....	274
Editing.....	274
Execution .....	275
Fields.....	275
Call Routing Protocol .....	276
Icon .....	275
Link Utilization Update Interval	276
Name .....	275
Packet Routing Protocol .....	276
Traffic Scale.....	275
Leaving .....	313
Naming Convention .....	307
Purpose.....	274
Reporting .....	275
subnetwork node .....	32
Sum .....	406
Sum of Squares .....	406
suspended applications .....	72
Switch .....	4, 21, 22, 174
Connectivity.....	174
Creating.....	174
Editing.....	175
Execution .....	175
Fields.....	177
Purpose.....	174
Reporting .....	177
switches.....	21
Switching Delay.....	4
<b>T</b>	
Table Distribution .....	321, 431
Fields	
Table Type	
Continuous .....	93
Discrete .....	93
Tag Window .....	408, 412
TCP/IP .....	27, 40, 318
Text Font.....	409
throughput rate .....	4
token bus .....	22
Token Passing.....	22
token ring .....	22
Tool	
Access Point.....	307
Background.....	306
Call Source.....	14
Connection .....	2, 14, 305
Link.....	14, 304
Node.....	14, 304
Palette.....	2, 8, 303
Selection.....	14, 305
Subnetwork .....	306
Topology.....	21
Trace .....	7, 9, 322
Traffic Files	
External.....	96
AutoBaseliner .....	96
Traffic Policing	
ATM.....	277
Frame Relay .....	278
Traffic Policing Algorithm .....	27
Traffic Shaping Algorithm.....	27
Traffic Source .....	23, 25
Application.....	23, 24, 279
Connectivity.....	279
Creating.....	279
Editing.....	279
Execution .....	280
Fields.....	281
Command Sequence .....	282
First Arrival.....	281

- Icon .....281
- Interarrival Time Distribution281
- Last Arrival .....282
- Max Arrivals .....282
- Name .....281
- Received Message Delay .282
- Received Message List ....282
- Schedule By .....281
- Purpose.....279
- Reporting .....281
- Call.....23, 283
  - Connectivity .....283
  - Creating.....283
  - Editing.....283
  - Execution .....283
  - Fields.....284
    - Call Routing Class .....285
    - Destination Type.....285
    - Duration .....284
    - First Arrival.....284
    - Icon .....284
    - Interarrival Time Distribution284
    - Last Arrival .....284
    - Name .....284
    - Priority .....285
  - Purpose.....283
  - Reporting .....284
- Message .....23, 286
  - Connectivity .....286
  - Creating.....286
  - Editing.....286
  - Execution .....287
  - Fields.....287
  - Purpose.....286
  - Reporting .....287
- Packet Flow.....288
  - Creating.....288
  - Purpose.....288
- Response .....23, 289
  - Connectivity .....289
  - Creating.....289
  - Editing.....289
  - Execution .....290
  - Fields.....290
  - Purpose.....289
  - Reporting .....290
- Session .....23, 291
  - Connectivity .....291
  - Creating.....291
  - Editing.....291
  - Execution .....292
  - Fields.....292
  - Purpose.....291
  - Reporting .....292
- TRAF LINK.....25
  - Purpose.....80
- Transit Network .....21, 23, 26, 32, 293
  - Compression or Encryption .....293
  - Interface Ports .....294
  - Link Behavior .....293
  - Link Level Flow Control .....293
  - Multicasting .....295
  - Protocol Layering .....293
  - Purpose.....293
  - Service Classes and Connection Types  
294
  - Session Routing .....295
  - Subnet Behavior.....293
  - TCP/IP over Frame Relay or ATM293
  - Traffic Constraints .....295
- Transit Networks.....293
- Transport & Answer Commands Reports342
- Transport Command .....358, 371
- Transport Message Command .24, 25, 68
  - Connectivity .....68
  - Creating.....68
    - Global Command.....68
    - Library Command.....68
    - Local Command.....68
  - Editing.....68
  - Execution .....69
  - Fields.....71
    - Destination Type.....71
    - Multicast List .....71
    - Random List.....71
    - Random Neighbor.....71
    - Weighted List.....71
  - Message Size Calculation .....71
  - Message Size Units .....71
  - Message Text .....71
  - Name .....71
  - Packet Routing Class .....71

Packetizing Time .....	71
Transport Protocol .....	71
Priority .....	71
Purpose.....	68
Reporting .....	70
Transport Protocol .....	296, 356, 358, 359
Connectivity .....	296
Creating.....	296
Editing.....	296
Execution .....	297
Fields.....	297
ACK Priority .....	299
End-to-End ACK Bytes .....	298
Error Control ACK .....	299
Flow Control Method.....	299
Name .....	297
Negotiated Packet and Window Sizes .....	299
Packet Data Bytes .....	297
Packet Overhead Bytes .....	297
Padded Packets .....	299
Retransmit Blocked Packets ..	299
Retransmit Time .....	298
Window Size.....	298
Purpose.....	296
Reporting .....	297
Transport Protocols.....	27
Triangular Distribution .....	428
Trigger .....	24
Trigger Events.....	300
Purpose.....	300
Trigger Rules .....	301

## U

Uniform Distribution .....	429
UniformInteger Distribution .....	429
UNIX .....	2, 7
User Defined Routing Tables .....	26, 27
User distribution .....	321, 431
User Distributions .....	29
Utilization by Application .....	341
Utilization by Protocol.....	341

## V

VC .....	50
View Menu .....	23, 311
view menu.....	22

Virtual Circuit	
Statistics .....	406
Burst Size .....	406
Frame Delay.....	406
virtual circuit.....	50
Virtual Circuits .....	23
virtual circuits .....	25, 40, 318
Voice.....	318
Voice Network.....	4

## W

Wait For Command .....	72
Connectivity.....	74
Creating.....	74
Global Command.....	74
Library Command.....	74
Local Command.....	74
Editing.....	74
Fields.....	74
Command Name .....	74
Get required msg text from command's input parameter	75
Only Accept Message Responding To This App .....	75
Required Message Text .....	75
Retain Previous Received Message If Any.....	75
Stop Waiting (and terminate app) after .....	75
Wait For Alarm.....	75
Wait for Expression .....	75
Purpose.....	72
Reporting .....	74
Stop Waiting .....	73
Troubleshooting .....	74
Wait For Alarm.....	73
Wait For Expression .....	73
Wait For Received Message .....	72
WAN.....	2, 4, 5, 22
WAN Cloud .....	21, 22
WAN Cloud Reports.....	341
WAN Clouds.....	23
WANs .....	40
warmup .....	28
Weibull Distribution .....	430
Weighted List.....	88

Index

Connectivity .....	88
Creating.....	88
Editing.....	88
Execution .....	89
Fields.....	89
Destination Node Name .....	89
Probability.....	89
Purpose.....	88
Reporting .....	89
Wide Area Network .....	21
window	
fixed .....	27
sliding.....	27
Workstation.....	4, 5
Write .....	24
Write Command.....	344, 347
Write File Command .....	76
Connectivity.....	76
Creating.....	76
Global Command .....	76
Library Command.....	76
Local Command.....	76
Editing.....	76
Execution .....	77
Busy The Node .....	77
Calculate Time Duration.....	77
Lock The File.....	77
Update File Information.....	78
Fields.....	78
Bytes To Write Calculation .....	78
Based On File Size.....	79
Based On Message Size .....	79
Probability Distribution .....	79
File Modification Method .....	78
Append.....	78
Replace.....	78
Update .....	78
File To Access .....	78
Name .....	78
Purpose.....	76
Reporting .....	78

X

X.25.....	3, 27, 40, 47, 318
-----------	--------------------