Univerza v Mariboru
Fakulteta za organizacijske vede

Dogodkovna simulacija sistemov

Miroljub Kljajić, Igor Bernik, Andrej Škraba

Delovna verzija 1.0, nerecenzirano interno gradivo

Kazalo

Predgovor

Pričujoče delo naj bi kandidatu omogočilo dostopnost preko svetovnega spleta ter s tem prispevalo k izboljšanju podajanja študijskega gradiva. Skripto je v postopku dopolnjevana zato so predlogi za izboljšave vedno dobrodošli.

Cilj predmeta je seznaniti slušatelje z možnostjo uporabe simulacijskih modelov pri preučevanju dinamike organizacijskih sistemov kot posledice poslovnih odločitev, organizacijskih sprememb in vplivov poslovnega okolja. Z opravljenimi vajami je slušatelj usposobljen, da na podlagi verbalnega opisa organizacijskega problema določi strukturo ter simulacijski model preučevanega organizacijskega sistema s pomočjo katerega preuči obnašanje ter vpliv povratnih zvez na dinamiko sistema. Pri oblikovanju modela so uporabljeni postopki metodologije sistemske simulacije kjer na podlagi ciljev, izhodišč ter verbalno podanega problema določimo blokovni diagram (strukturni model) s pomočjo katerega določimo povratne zanke, elemente stanja ter elemente spremembe stanja sistema. Model je podan z matematičnim opisom ter simulacijskim programom. Z izborom kriterijskih funkcij ter eksperimentiranjem pristopimo k iskanju rešitve zastavljenega problema. Predstavljeni način raziskovanja organizacijskih sistemov omogoča slušateljem razvoj intelektualnih navad ter vsestransko in racionalno proučevanje in razumevanje organizacijskih sistemov in problemov.

Avtorji

1 Uvod v dogodkovno simulacijo sistemov

Simulacija kot način reševanja pomembnih in kompleksnih problemov predstavlja zelo staro raziskovalno disciplino. Že vojskovodje in vladarji v letih pred našim štetjem so na vojaških vajah uprizorili različne možno strategije nasprotnika ter svoje odgovore. Ves čas razvoja družbe je bila vojaška simulacija (znana pod nazivom vojaške vaje) pomemben dejavnik urjenja in pridobivanja novih znanj ter "izkušenj". Tudi v drugih panogah, ki so bile pomembne in zapletene ter med seboj povezane, so bolj intuitivno kot znanstveno uporabljali metodo simulacije v širšem pomenu. Z pojavom računalnika postaja simulacija znanstvena disciplina oz. del sistemskega pristopa. Področje uporabe simulacijskih modelov je široko ter sega na vsa področja znanosti od podpore odločitvenimi procesov v organizacijskih znanostih do simulacije najrazličnejših tehničnih sistemov. Uporaba simulacije se tako širi na področja industrije, ekonomije, transporta, urbanizma, osvajanja vesolja, načrtovanja, ter nenazadnje vojski.

Namen simulacije je analiza odzivov nekega sistema v prihodnjem času ali pa povečanje razumevanja obravnavanega sistema. Na ta način prihranimo stroške eksperimentiraja na realnih sistemih ter se izognemo morebitnim nevarnostim, ki so s tem povezane.

Simulacijo uporabljamo kadar je problem, ki ga rešujemo, kompleksen in ga ne moremo rešiti z drugimi metodami ali pa pri pojavih pri katerih ne smemo ali pa ne moremo neposredno pristopiti k izvajanju eksperimenta.

Simulacijski modeli pravzaprav niso izključno vezani na računalnik, vendar pa je uporaba računalniške simulacije tako razširjena, da je to postala sinonim za simulacijo kot tehniko reševanja problemov. Povdariti moramo, da je računalniški oziroma simulacijski model nekega sistema pravzaprav le mnogo bolj precizen in razširjen miselni model oziroma predstavlja združitev in poenotenje modelov, ki si jih več ljudi oblikuje o nekem proučevanem sistemu ali delih tega sistema. Računalnik deluje le kot nekakšen informacijski "ojačevalnik" človekovega poznavanja sistema.

Za širšo uporabo simulacije v poslovnih okoljih so potrebne preproste in učinkovite rešitve. Delovati morajo z realnimi podatki, hkrati pa za uspešno delo ne smejo zahtevati previsokega nivoja specialnih računalniških znanj. To je posebej pomembno zato, ker so podatki poslovne simulacije namenjeni predvsem managerjem. Ti se morajo odločati hitro, njihove odločitve pa morajo temeljiti na zanesljivih in svežih informacijah. Uporaba simulacije v poslovnih sistemih še ni vsakdanji pojav, vendar pa njena uporaba stalno narašča.

Pospešeno uvajanje simulacije poslovnih sistemov je posledica izboljšane komunikacije človek-stroj, ta pa posledica približevanja orodij managerjem. Poslovni modeli niso enostavnejši, kot so bili desetletje nazaj. Prav tako ni nikakršnih razlogov, da bi se managerji želeli naučiti simulacijskih tehnik za zabavo. Pa vendar je revolucija v informatiki približala računalnike uporabnikom. Zaradi grafičnih uporabniških vmesnikov modele lažje gradimo. Simulira in eksperimentira lahko vsakdo z osnovnim znanjem informatike, preizkuša različne alternative in scenarije ter preučuje rezultate obnašanja sistemov. Poslovna simulacija je tako dejansko prenesena iz visoko specializiranih laboratorijev na mize uporabnikov, ki imajo željo in potrebo po njeni uporabi. Izkušnje poudarjajo (npr. Girbone et al., 1995), da v novi situaciji, v različnih industrijah, za različne namene, v različnih pogojih lahko rešujemo obstoječe probleme in se hkrati izognemo novim potencialnim problemom ravno z uporabo simulacijske tehnike.

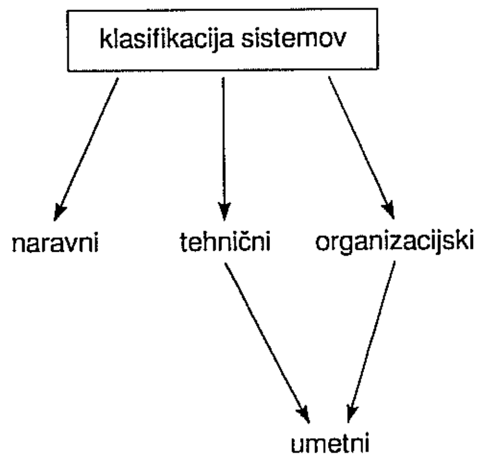
Različne raziskave v svetu in literatura (Dijk et al., 1996; Kljajić et al., 1996; Saltman, 1997; Tung, 1987) ugotavljajo, da kombinacija simulacije in sistemov za podporo odločanju omogoča kakovostnejše odločitve. Simulacija, ki z animacijo prikazuje delovanje modeliranega sistema, udeležencem pomaga, da se hitro učijo specifike delovanja prikazanega sistema. V raziskavi (Dijk et al., 1996) je potrjeno, da zaradi animacije odločevalci bolje razumejo rezultate simulacije. Če tudi simulacijski model potrdi mnenja o posledicah uvedbe neke spremembe v sistem, so odločevalci bolj zavzeti pri utemeljevanju predlagane spremembe. Simulacija in animacija motivirata odločevalce pri iskanju novih rešitev, saj jim omogočata preskus variant, ki jih v realnem sistemu ne morejo izvesti.

Dogodkovna simulacija omogoča določanje izkoristka obstoječe tehnologije in ugotovitev ozkih grl, ugotavljanje odvisnosti časa za dobavo izdelkov v funkciji povpraševanja, izdelavo modela za operativno planiranje proizvodnje ter analizo obnašanja sistema v prihodnosti (Kljajić, 1996).

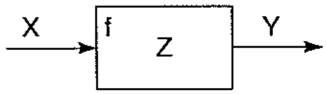
Kompleksnost gradnje modela in izvajanje simulacijskih eksperimentov je v preteklosti omejevala uporabo simulacije v poslovnih sistemih. Osnovna ideja vizualnega interaktivnega modeliranja je, da model razvijamo v vizualnem interaktivnem okolju in na isti način izvajamo simulacijo, kar olajša gradnjo modela in izboljša percepcijo o delovanju sistema.

1.1 Modeliranje in simulacija

Sistem je množica elementov ali enot, ki so povezane v določeno celoto. Vsakemu elementu pripadajo določene lastnosti oz. atributi in dejavnosti oz. aktivnosti.



Na splošno opisujejo sistem:



- vhodne spremenljivke $X = \{X_i\}; i = 1,2,3,...,m$
- stanja sistema $Z = \{Z_j\}; j = 1,2,3,...,n$
- izhodi iz sistema $Y = \{Y_r\}; r = 1,2,3,...,l$

Primer:

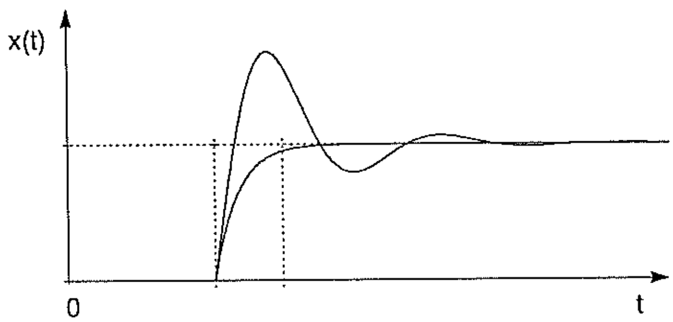
sistem	elementi	lastnost	aktivnosti
javni promet	vozila	hitrost, razdalja	vožnja zaviranje
veleblagovnica	kupci	nakup	plačevanje
banka	poslovni partnerji	stanje	polog, dvig
komunikacijski sistem	sporočila	dolžina, prioriteta	prenos

Stanje sistema je vrednost spremenljivk stanja ob določenem času. Stanje sistema opišemo z elementi sistema; njihovimi lastnostmi in aktivnostmi.

Proces je sprememba stanja sistema pod vplivom vhodnih spremenljivk ali notranjih dogodkov v sistemu.

Obnašanje sistema

Odziv (reakcija) sistema na vhodne signale (dražljaje) imenujemo OBNAŠANJE SISTEMA.



stanja dinamičnega sistema pri obnašanju

1.2 Definicija modela

Model sistema je poenostavljena ponazoritev (koncept) realnega sistema.

Sistemska simulacija je način reševanja problemov s pomočjo eksperimentiranja na računalniškem modelu z namenom, da analiziramo delovanje celote ali posameznih delov sistema pri določenih pogojih.

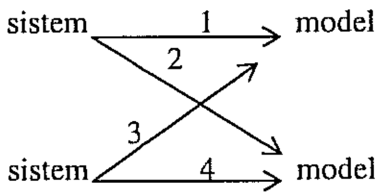
Simulacija je dinamična ponazoritev obnašanja modela v naslednje namene:

1. Opis sistema (delni ali v celoti).
2. Razlaga obnašanja sistema v preteklosti.
3. Predvidevanje obnašanja sistema.
4. Razumevanje zakonitosti sistema.

Sistemska paradigma simulacije: Potek modeliranja pri računalniški simulaciji

- definicija problema
- določitev ciljev
- osnutek študije
- formiranje matematičnega modela
- zapis računalniškega programa
- validacija modela
- priprava eksperimenta (simulacijskih scenarijev)
- simulacija in analiza rezultatov

1.3 Odnosi med sistemom in modelom



1. Sistem determinističen model determinističen
   Preprosti mehanski modeli $\ddot{x}+B\dot{x}+Cx=U(t)$.

2. Sistem determinističen model stohastičen
   Način poenostavitve zapletenih funkcij (metoda Monte Carlo).
3. Sistem stohastičen model determinističen
   Kongruenčni generatorji naključnih števil.

4. Sistem stohastičen, model stohastičan
   Kompleksni organizacijski sistemi – reševanje s pomočjo sistemske simulacije.

1.4 Modeliranje in simulacija

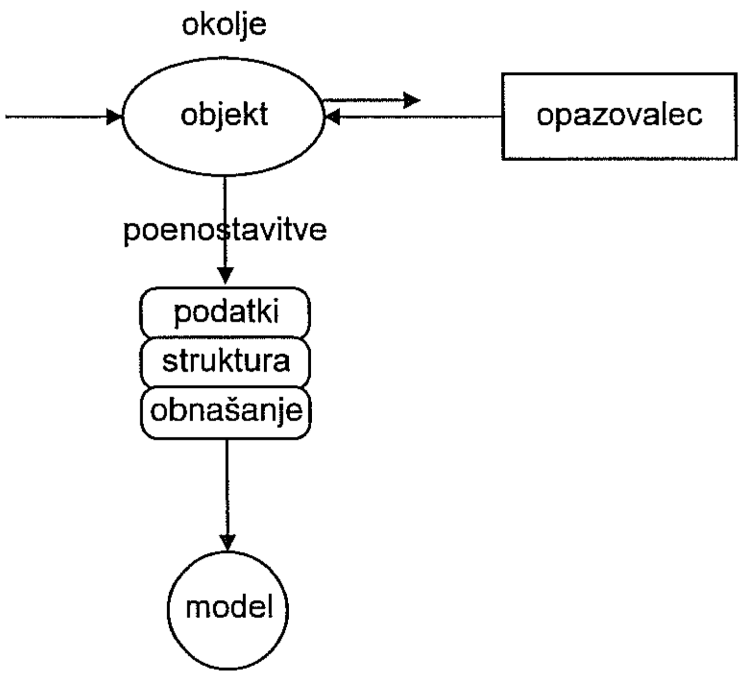
Modeliranje predstavlja relacijo med simuliranim sistemom in modelom, simulacija pa relacijo med modelom in računalniškim procesom.
Oglejmo si nekaj definicij simulacijskih modelov:

X simulira Y če in samo, če:
a) X in Y sta sistema
b) Y predstavlja simulirani sistem (sistem)
c) X predstavlja poenostavitev simuliranega sistema (model)
d) veljavnost X v odvisnosti od Y ni nujno popolna

Proces generiranja obnašanja modela imenujemo simulacija. Simulacija je eksperimentiranje na modelu.

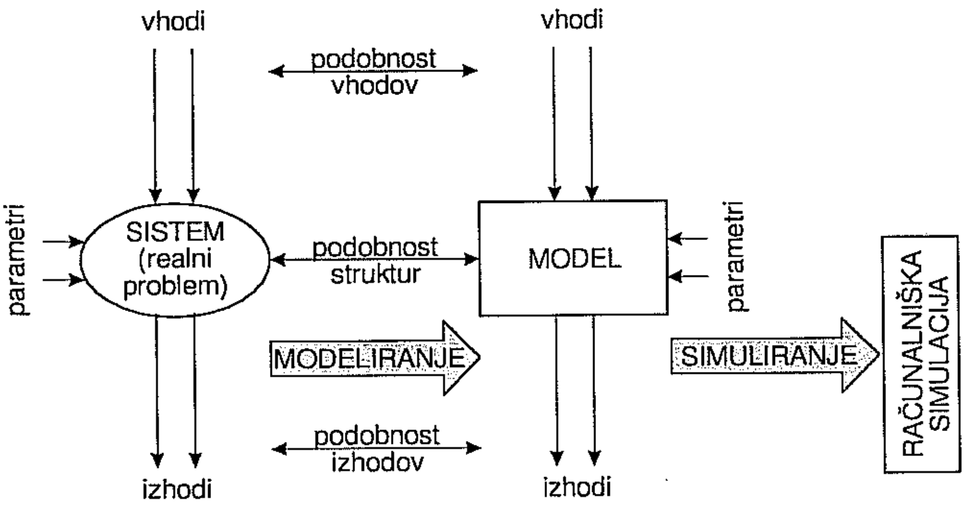
1.5 Metodologija modeliranja

Ko opredelimo problem, ki ga želimo študirati, in opredelimo, na kakšna vprašanja želimo dobiti odgovore, se lotimo postavljanja modela. Ugotoviti moramo pomembne spremenljivke in njihove medsebojne povezave, kar je obenem poglavitni del naloge. Iz množice podatkov moramo izločiti tiste, ki so dominantni za izbrani vidik proučevanja sistema. Tu nekega pravila ni. Izjeme so naravni sistemi, kjer veljajo zakoni ohranitve energije in zveznosti. V drugih primerih pa skušamo spremenljivke tako izbrati, da jih lahko ustrezno interpretiramo. Dobro nam pri tem služijo izkušnje in intuicija. Potek procesa modeliranja kaže slika 1



Slika 1. Potek procesa modeliranja.

Raziskovalec na podlagi ciljev in poznavanja sistema napravi potrebne poenostavitve, določa strukturo sistema, zbira podatke in na podlagi tega v okviru obstoječe teorije zgradi ustrezen model, s pomočjo katerega lahko študira lastnosti realnega objekta, ali pa izbira ustrezno strategijo vplivanja na realni sistem.



Po načinu opisa in zaporedju nastanka ločimo naslednje modele:

- Verbalni, ko v naravnem jeziku opišemo zakonitosti, ki veljajo med elementi problema, ki ga modeliramo. Na primer Newtonov zakon akcije in reakcije, ki je podlaga za model sistema na sliki 3.21. Verbalni model je lahko faktualni opis dogodkov, ki jih ne moremo vnaprej formalizirati. Za takšne modele je bistvenega pomena sintaktična, v okviru leksike in dejstva, semantična ali interpretativna, in pragmatična, oziroma vrednostno uporabna korektnost.
- Fizični modeli so ponavadi analogne ali v istem materialu narejene miniaturne podobe, ki so zaradi obstoja analogij med oblikami in obnašanjem zelo koristne za raziskave, ki bi sicer na originalu bile drage in nevarne.
- Matematični ali formalni modeli so abstraktni in najbolj precizni opisi nekega objekta. Matematični model brez predhodnih faz v opisu nekega objekta je brez pomena.

1.6 Odnosi med modelom in originalom

Kot smo že videli, model odraža določene lastnosti originala. Odnos modela in originala je lahko homomorfen ali izomorfen. Homomorfni modeli predstavljajo poenostavljanje v odnosu do originala, saj model opisuje le bistvene lasnosti originala. To pomeni, da so zaključki, ki jih dobimo na takšnem modelu, omejeni. Do poenostavitve ponavadi pridemo z združevanjem več lastnosti sistema v eno reprezentančno spremenljivko ali pa z opuščanjem nepomembnih spremenljivk originala. Pri kompleksnih sistemih je normalno, da uporabljamo poenostavljene modele. Izomorfnost med modelom in originalom pomeni, da se model in original istovetno funkcionalno obnašata na relaciji vhod - izhod iz sistema, ne glede na njuno bistvo

Po stopnjah podobnosti modela in originala ločimo:

- zunanjo podobnost,
- strukturno podobnost,
- funkcionalno podobnost.

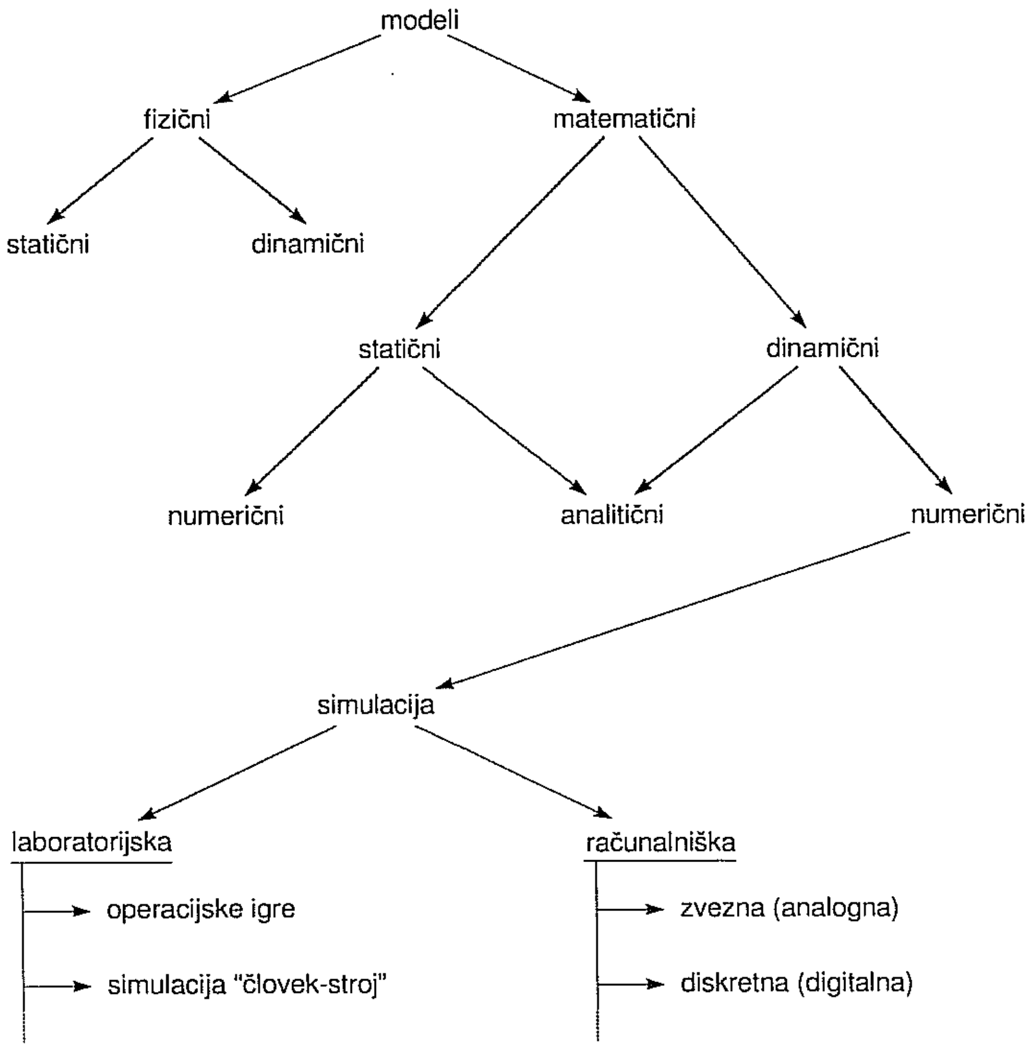
1.7 Sistem, model in simulacija

Po načinu modeliranja ločimo:

- metodo analogije,
- metodo črne škatle (black box),
- metodo sive škatle (gray box),
- metodo prozorne škatle (white box).

1.8 Klasifikacija modelov

Modele lahko klasificiramo na več načinov. Ena od možnih klasifikacij (Forrester, 1961) je prikazana na naslednji sliki in ustreza klasifikaciji sistemov, ki smo jo podali v prvem poglavju.



Izbira modela je odvisna od sistema, ki ga proučujemo. Gornja slika - shema razčlenjuje fizične in abstraktne modele kot možen način proučevanja sistemov. Opozorimo naj še enkrat: slehernemu modelu predhodi teorija, ki je nujno kvalitativna. V odvisnosti od njene abstrakcije lahko govorimo o eksaktnih ali verbalnih teorijah, odvisno od pojava, ki ga želimo interpretirati. Fizični modeli so ponavadi poenostavljeni in zmanjšani realni sistemi, katerih lastnosti proučujemo. Fizični modeli nazorno predočajo obnašanje realnega sistema, predvsem njegovih pomembnejših lastnosti v določenem delovnem okolju in pri določenih pogojih. Statični fizični modeli so npr. prikaz urbanističnih in arhitektonskih rešitev v obliki maket, katere nam omogočajo vizualne predstave določene prostorske oblike. Dinamični fizikalni modeli so npr. aerodinamični tuneli za proučevanje lastnosti letal, hidrodinamični kanali za proučevanje hidrodinamičnih lastnosti ladij itn.

Kot dinamični fizični model bi lahko vzeli eksperimentalne oblike, kjer proučujemo vpliv ekoloških sprememb na različne vrste flore in favne in iščemo nove variacije z umetno mutacijo, da bi dobili vzorce, ki se najbolj prilagajajo novim pogojem, itn. Dobre lastnosti te vrste modelov so v tem, da so jasni in pregledni in v precejšnji meri ustrezajo realnemu sistemu. Slaba stran je ta, da so veliki, nefleksibilni in pogosto ne pokažejo vzročno-posledične odvisnosti med posameznimi pojavi in spremenljivkami.

1.9 Matematično modeliranje

Matematični model je abstraktna ponazoritev določenega sistema in nam rabi za njegovo proučevanje.

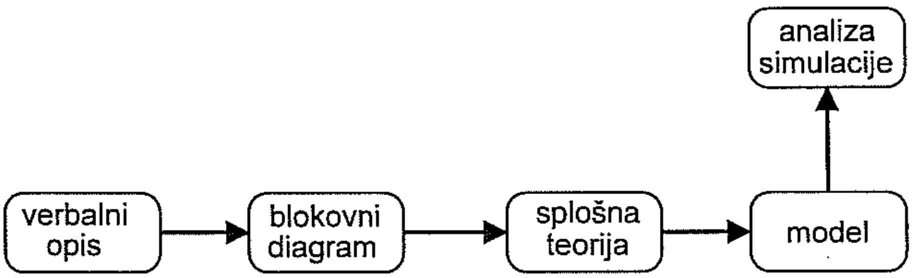
Če za opis realnega sistema uporabimo kakšen formalni jezik, temu rečemo matematični model. Na osnovi matematičnega modela lahko izvedemo zaključke o karakteristikah in obnašanju sistema.

Matematični modeli so v večji ali manjši meri homomorfni realnemu sistemu. Od stopnje ustreznosti modela in sistema je odvisno, koliko se lahko zanesemo na rezultate in zaključke, do katerih smo prišli s proučevanjem modela. Tu je treba opozoriti, da kriterija za preverjanje, ali je model adekvaten ali ne, ni. Edini zanesljiv kriterij je praksa in izkušnja.

Vrste matematičnih modelov so lahko zelo različne: npr. grafi, tabele, enačbe, logični simboli itn., ki ponazarjajo določeno stanje sistema in njegovo obnašanje. Zaradi svoje preciznosti v izražanju in možnosti za analizo prihodnjega obnašanja sistemov v kvantitativni obliki so matematični modeli najzanimivejši za teorijo sistemov. Z njihovo pomočjo analiziramo obnašanje sistema in se odločamo o vrsti upravljanja. Vedno pa se moramo zavedati, da pogosto ni enostavno, ali celo ni možno, nekemu realnemu sistemu najti ustrezen matematičen model.

1.10 Metodologija računalniške simulacija

Analitično rešitev diferencialnih enačb, ki opisujejo nek dinamičen sistem, lahko poiščemo le v najbolj enostavnih in idealiziranih primerih. V bolj kompleksnih sistemih, kjer nastopajo zapleteni sistemi diferencialnih enačb, pa moramo za njihovo reševanje uporabiti numerične metode. Ena od njih, prav gotovo najbolj razširjena, je računalniška simulacija. V zadnjih dveh desetletjih so bili razviti različni problemsko orientirani simulacijski jeziki. Modeliranje seveda zahteva tudi dobro poznavanje problema, ki ga želimo proučevati s pomočjo računalniške simulacije. Potek modeliranja prikazuje naslednja shema



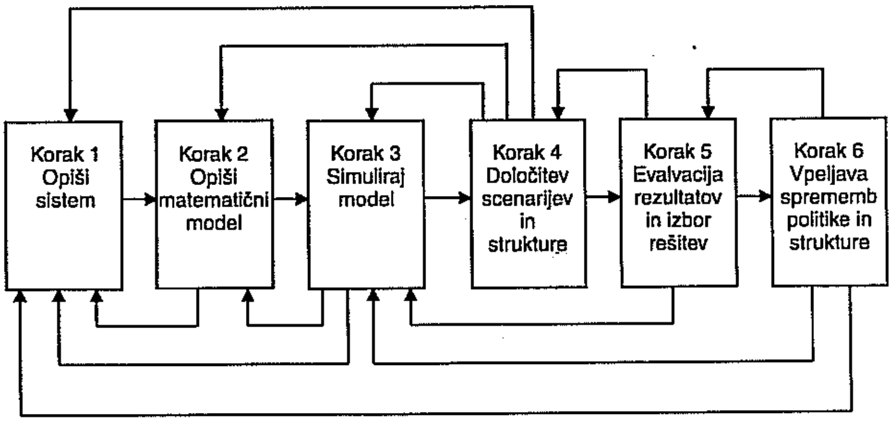
Slika 2. Potek modeliranja v računalniški simulaciji.

a) Pri definiciji problema opredelimo raven in cilj modeliranja, obseg obravnavanega sistema, interakcije z okoljem itn.
b) Sledi določitev spremenljivk, povratnozančnih zvez in interakcij med spremenljivkami in med deli obravnavanega sistema. Te zveze lahko ilustriramo z bločnim diagramom, ki v grobem definira dinamiko sistema in vzročno-posledične odnose v sistemu.
c) Blok splošna teorija pomeni analizo problema v splošnejšem okviru, ki vzpostavlja zvezo med konkretnim problemom in konkretnimi rešitvami s splošnejšega vidika.
d) Sledi matematično detajliranje posameznih delov ali pa zapisov z ustreznimi enačbami modela, primernimi za izbrani simulacijski jezik. Vse korake ponavljamo, dokler ne dobimo zadovoljive rešitve.

Simulacijski modeli niso pravzaprav izključno vezani na računalnik, toda uporaba računalniške simulacije je tako velika, da je to sinonim za simulacijo kot tehniko reševanja problemov. Poudariti moramo, da je računalniški oziroma simulacijski model nekega sistema pravzaprav le mnogo bolj precizen in razširjen miselni model oziroma združitev in poenotenje modelov, ki si jih več ljudi postavi o nekem sistemu ali delih tega sistema. Računalnik deluje le kot nekakšen informacijski "ojačevalnik" človekovega poznavanja sistema.

Obstaja več enakovrednih načinov ponazoritev sistema, ki so primerni za računalniško simulacijo. V simulaciji poslovnih sistemov se je uveljavila metodologija sistemske dinamike (SD), ki jo je predlagal J. Forrester (Forrester,1961). (V resnici, ko jo je vpeljal, jo je imenoval metodologija industrijske dinamike, vendar je razvoj klasične industrije v smeri informatizacije in robotizacije podjetij v smeri globalnih poslovnih sistemov zahteval razširitev naziva). In blokovni diagrami za dogodkovno simulacijo.

Metoda sistemske dinamike ni le pisanje enačb gibanja poslovnih procesov, temveč tudi celovita metodologija reševanja dinamičnih problemov. Njen postopek kaže slika 3 (Forrester,1994)



Slika 3 Postopek reševanja problemov v metodologiji sistemske dinamike.

Pri tem prvi korak pomeni definicijo problema, ciljev in izhodišč. Ta del je lahko katerakoli metodologija sistemskega pristopa. Drugi korak pomeni zapis problema v simulacijski model. Ostali koraki so razvidni iz diagrama. Puščice kažejo postopek reševanja po korakih, s tem da naslednja faza lahko iterativno vpliva na predhodno fazo. Z drugimi besedami, posamezne faze so soodvisne. V primeru, da v drugi fazi ni možno problema prevesti na simulacijski model, se zadovoljimo s kvalitativno analizo strukture sistema na blokovnem diagramu. V tem primeru temu grafu lahko rečemo tudi kognitivna mapa ali pa diagram vpliva. V vsakem primeru lahko udeleženci reševanja problema racionalno razpravljajo o posledicah odločitve.

Ostane nam, da pokažemo metodologijo, ki smo jo razvili za potrebe biokibernetike (Kljajić, 1989) in sistemov za podporo odločanja v poslovnih sistemih (Kljajić, 1994) in je pravzaprav razdelana metodologija sistemske dinamike.

> Pričujoča metodologija obravnava obnašanje integralnega simulacijskega sistema za pomoč pri poslovnih odločitvah in analizi obnašanja organizacijskih sistemov. Namen metodologije je:
>
> - spoznavanje obnašanja integralnega simulacijskega sistema za pomoč pri poslovnih odločitvah, strateškem planiranju in analizi organizacijskih sistemov pri različnih kriterijih in scenarijih:
> - izboljšanje procesa planiranja in odločanja
> - pridobitev novih znanj o obnašanju in upravljanju kompleksnih sistemov
> - vzgoja strokovnega kadra za planiranje in vodenje podjetja.

1.11 Postopki modeliranja in simulacije

1. Definicija problema kot sistema
2. Izbor izhodišča
3. Določitev ciljev in sodil
4. Razmejitev sistema in okolja: določitev vhodnih in izhodnih funkcij
5. Določitev elementov sistema
6. Določitev povezav med elementi (struktura)
7. Opredelitev lastnosti in aktivnosti zvez ter vhodnih in izhodnih funkcij

8. Opredelitev modela sistema
9. Validacija modela
10.Priprava scenarijev
11.Analiza rezultatov in izbor rešitve

Literaturo, kjer smo podrobno opredelili opis posameznih točk, najdete v (Kljajić, 1989 in Kljajić, 1994). Pri večini simulacijskih projektov je delo timsko, zaradi tega je potrebno nameniti veliko pozornost tudi predstavitvi in dokumentiranju spoznanj v procesu iskanja rešitev. Dokumentacija simulacijskega projekta naj bi glede modela in simulacije vsebovala:

1. Neformalni opis modela in utemeljitev predpostavk
2. Formalni opis - blokovni diagram in matematični model
3. Simulacijski program
4. Analizo rezultatov simulacijskih eksperimentov
5. Povezavo predpostavljenega modela s podobnimi
5. Določitev uporabnosti rezultatov ter analizo stroškov/koristi

2 Dogodkovna simulacija

Poudarek je na dogodkih, ki vplivajo na sistem. Dogodki lahko:

- povzročijo spremembo vrednosti neke spremenljivke sistema
- sprožijo ali prekinejo sistemsko spremenljivko
- aktivirajo ali prekinejo določen proces

Diskretne dogodke lahko obravnavamo z dveh vidikov:

Pri elementarni orientiranosti (particle oriented) predstavljajo izhodišče za simulacijsko analizo elementi sistema.
Pri dogodkovni orientiranosti (event oriented) predstavljajo izhodišče za simulacijsko analizo dogodki, ki nastopajo pri procesu obravnavanega sistema.

2.1 Predstavitev časa

Pri simulaciji diskretnih sistemov predstavimo čas z interno simulacijsko uro $t_z = 0$. Razmerje med simulacijskim časom in realnim časom je odvisno od narave obravnavanega sistema.

Generiranje časov prihodov elementov (transakcij) je odvisno od sistemskih pogojev.

Načini oblikovanja vrst (queuing disciplines)

Osnovni pojmi so:

1. Zaporedje prihajanja elementov (transakcij) (arrival patterns).
2. Obdelava sistemskih elementov (transakcij) (service process).
3. Načini oblikovanja vrst (queuing disciplines)

Obdelavo (strežbo servis) sistemskih elementov (transakcij) opisujemo s časom obdelave (časom servisa) (service time) in s kapaciteto obdelave (kapaciteto strežnega mesta, servisno kapaciteto).

Čas obdelave (čas strežbe, servisa) je čas potreben za obdelavo (čas strežbe, servis) posameznega sistemskega elementa (transakcije, dinamične entitete). Kapaciteta obdelave (kapaciteta strežnega mesta servisa) predstavlja število sistemskih elementov (transakcij), ki jih lahko obdelujemo istočasno.

Pri modeliranju sistema moramo podati verjetnostne porazdelitve časov med prihodi posameznih elementov (transakcij) in časov obdelave (strežbe).

2.2 Disciplina vrste

1. pravilo: FIFO (Prvi-Noter, Prvi-Ven, First-In, First-Out)
2. pravilo: LIFO (Zadnji-Noter, Prvi-Ven, Last-In, First-Out)
3. pravilo: naključno

2.2.1 Porazdelitev časov med prihodi

Prihode posameznih elementov (transakcij) v sistem običajno opišemo s časi med prihodi. V realnih sistemih srečamo neomejeno število različnih porazdelitev prihodov. S teoretičnimi porazdelitvami opišemo realne porazdelitve. Pri tem gre seveda le za bolj ali manj posrečen približek dejanskih porazdelitev. Najpogostejše teoretične porazdelitve, ki se uporabljajo pri simulacijah so:

- enakomerna porazdelitev
- eksponentna porazdelitev
- normalna porazdelitev
- Erlangova porazdelitev

Za opisovanje prihodov elementov (transakcij, strank) v sistem uporabljamo dva parametra:

T_a ... časovni interval med dvema zaporednima prihodoma

$\lambda = \frac{1}{T_a}$ ... hitrost prihodov

2.2.2 Poissonova porazdelitev

Verjetnost, da je čas od izhodišča do prvega popolnoma naključnega prihoda (Poissonov proces) enak t, (vrednost med t in $t+dt$) je podana z eksponentnim zakonom. Gostota verjetnosti je podana s funkcijo:

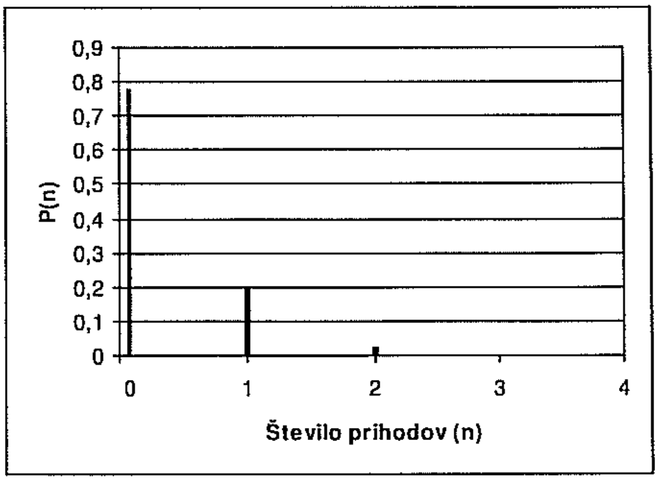
$$f(t) = \lambda e^{-\lambda t} \; ; \; t \geq 0 ,$$

ki predstavlja eksponentno distribucijo.

Porazdelitev števila prihodov znotraj konstantnega časovnega intervala t je podana s Poissonovo porazdelitvijo:

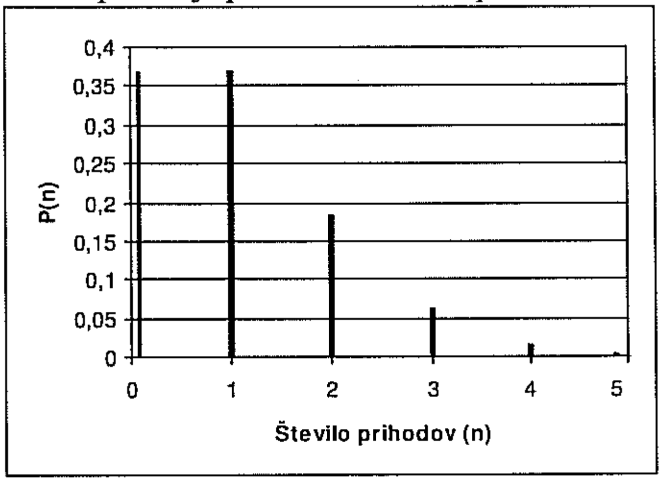
$$P(n) = \frac{(\lambda t)^n e^{-\lambda t}}{n!} \; ; \; n = 1,2,3,...$$

Slika 4 prikazuje primer Poissonove porazdelitve za λ=0,25.



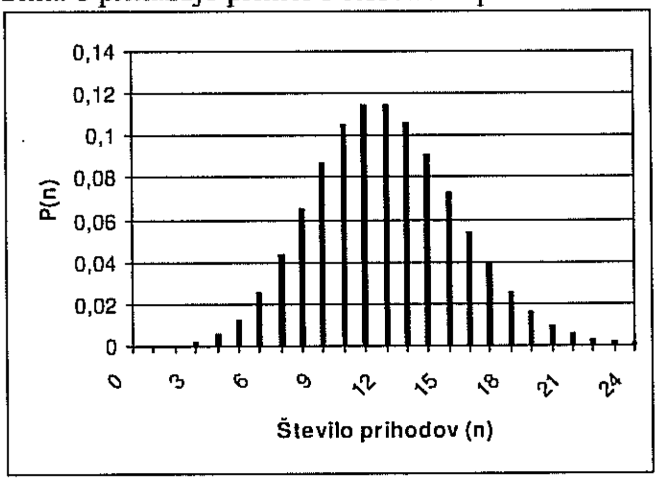
Slika 4: Poissonova porazdelitev za λ=0,25

Slika 5 prikazuje primer Poissonove porazdelitve za λ=1.



Slika 5: Poissonova porazdelitev za λ=1

Slika 6 prikazuje primer Poissonove porazdelitve za λ=12.



Slika 6: Poissonova porazdelitev za λ=12

3 Modeli strežbe

Modeli strežbe. Čas prihoda in čas strežbe. Disciplina vrste.
Generiranje časa prihoda in strežbe; enakomerna, eksponentna, Erlangova, normalna in numerična porazdelitev

Simulacija vrst in strežbe

Osnovne karakteristike sistemov množične strežbe so:

- porazdelitev časov prihodov strank
- porazdelitev časov strežbe
- število strežnih mest
- kapaciteta sistema strežbe
- disciplina vrste
- število stopenj strežbe

Simbolična oznaka sistema strežbe je A/B/X/Y/Z

Kjer pomenijo oznake naslednje:

A – porazdelitev časov prihodov strank (transakcij)
B – porazdelitev časov strežbe strank
X – število paralelnih strežnih mest
Y – omejitve kapacitete strežbe
Z – disciplina vrste

Tabela 1 prikazuje oznake osnovnih značilnosti sistemov strežbe.

Karakteristika	Oznaka	Pomen oznake
porazdelitev časov prihodov (A)	D	deterministična
	M	eksponentna
	E_k	Erlangova vrste *k*, *(k=1, 2, ...)*
	GI	splošna, neodvisna
distribucija časov strežbe (B)	D	deterministična
	M	eksponentna
	E_k	Erlangova vrste *k*
	G	splošna
število strežnih mest (X)	1, 2, ...	
kapaciteta sistema strežbe (Y)	1, 2, ...	
disciplina vrste (Z)	FIFO	prvi prispe, prvi postrežen
	LIFO	zadnji prispe, prvi postrežen
	SIRO	naključni vrstni red strežbe
	PRI	strežba glede na prioritete
	GD	splošni vrstni red

Tabela 1: Oznake osnovnih značilnosti sistemov strežbe

Primer: M/D/1/5/PRI

Sistem strežbe opredeljuje eksponentni čas prihoda strank, determinističen čas strežbe z enim strežnim mestom s skupno kapaciteto pet ter prioritetnim načinom strežbe.

Parametri, ki določajo lastnost strežnega kanala

L – povprečno število strank (transakcij) v sistemu
L_q – povprečno število strank (transakcij) v vrsti
W – povprečen čas, ki ga transakcija prebije v sistemu
W_q – povprečen čas, ki ga transakcija prebije v vrsti

Relacije med količinami, ki opredeljujejo uspešnost strežbe

parametri:

λ ~ hitrost prihoda transakcij (transakcije/enota časa)
μ ~ hitrost strežbe ~ (transakcije/enota časa) primer za en strežnik
μ' ~ $m*\mu$ ~ za več-mestno strežbo
$\rho = \frac{\lambda}{\mu}$ ~ faktor izkoriščenosti sistema, v primeru, da je $\rho > 1$ je sistem obremenjen
$P(t)$ ~ verjetnost, da je čas strežbe $> t$
$P_q(t)$ ~ verjetnost, da je čas čakanja v vrsti do pričetka strežbe > t

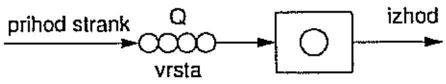
Tabela 2 prikazuje relacije med količinami, ki opredeljujejo uspešnost strežbe.

Relacija	Področje uporabe
$W = W_q + \frac{1}{\mu}$	splošno
$W = \frac{L}{\lambda}$	Little-ove formule (splošne)
$W = \frac{L_q}{\lambda}$	
$L = L_q + (1 - p_0)$	za enokanalne sisteme
$L = L_q + \frac{\lambda}{\mu}$	za vse sisteme kjer veljajo Little-ove formule
$p(0) = 1 - \frac{\lambda}{\mu}$	za enokanalne sistem za katere veljajo Little-ove formule
$W_q = \frac{L}{\mu}$	za vse sisteme M/M/1

Tabela 2: Relacije med količinami, ki opredeljujejo uspešnost strežbe

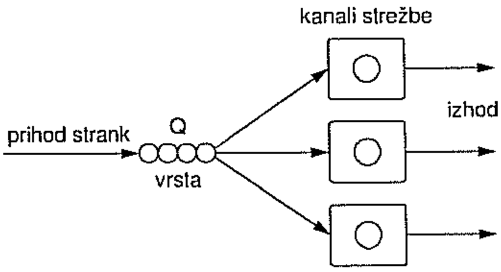
3.1 Tipi strežbe in vrst

Slika 1 prikazuje enostavni sistem strežbe z eno vrsto ter enim strežnim mestom.



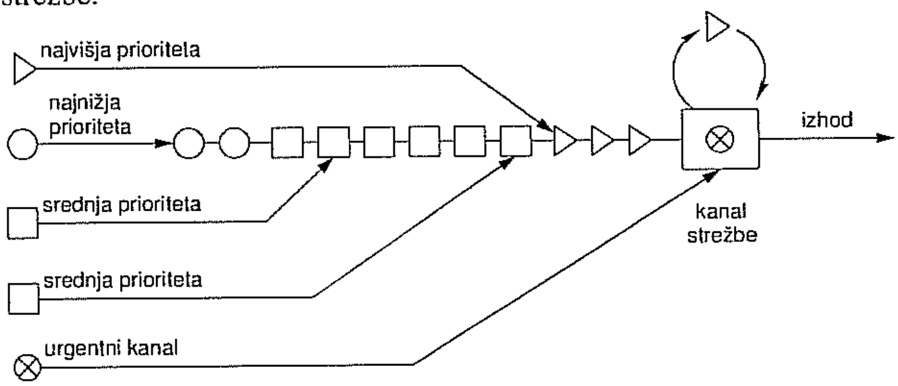
Slika 1: Ena vrsta, eno strežno mesto

Slika 2 prikazuje enostavni sistem strežbe s strežnim mestom, ki ima kapaciteto > 1 ter eno vrsto.



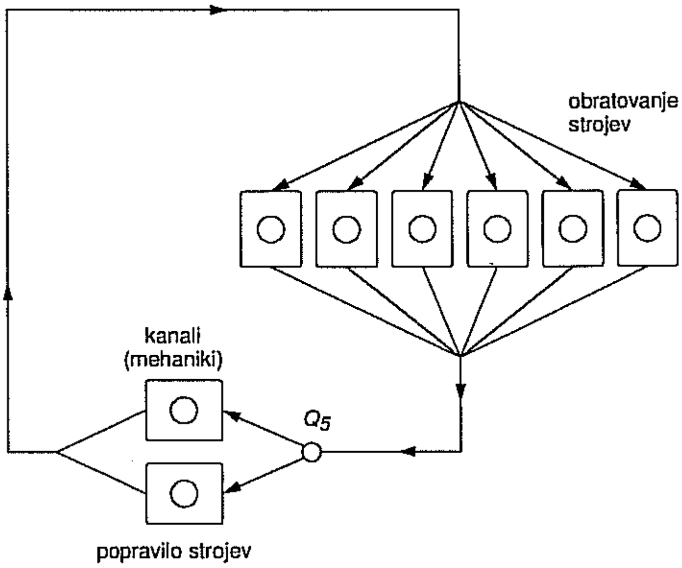
Slika 2: Enostavni sistem strežbe

Slika 3 prikazuje sistem strežbe strank s prioriteto. Različne prioritete vplivajo na način strežbe.



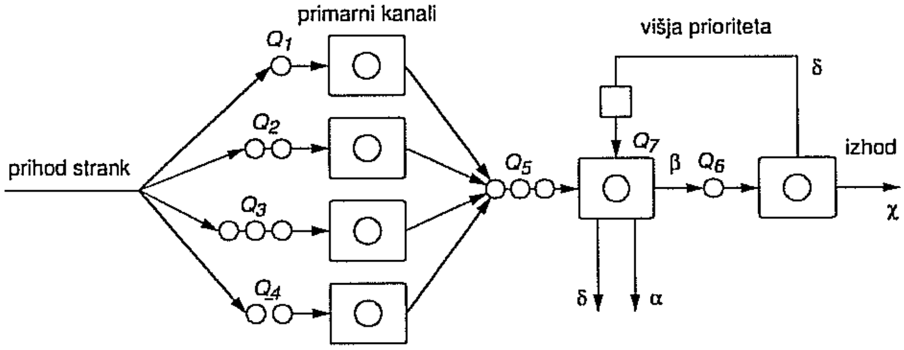
Slika 3: Strežba strank s prioriteto

Slika 4 prikazuje zaprti sistem strežbe, npr. vzdrževanje strojev.



Slika 4: Zaprti sistem strežbe

Slika 5 prikazuje bolj kompleksen, večkanalni sistem strežbe.



Slika 5: Večkanalni sistem strežbe

4 Verjetnostne osnove simulacije

4.1 Stohastična spremenljivka in verjetnostna funkcija.

Kot smo v uvodu povdarili za potrebe preučevanja kompleksnih organizacijskih sistemov potrebno je njihovo obnašanje opisati s stohastičnimi funkcijama. Bodisi da je znanje o sistemu nepopolno in ga moramo opisati kot naključno z ustrezno distribucijsko funkcijo, bodisi da je opis sistema zapleten in ga je treba aproksimirati s stohastično funkcijo bodisi da je njegovo obnašanje stohastično. Ker je računalniška simulacija eksperimentiranje na računalniku potem je vhode in procese v modelu sistema potrebno generirati naključne dogodke. Spremenljivki, katera predstavlja izhod naključnega dogodka rečemo stohastična spremenljivka. Dogodkom ki jih opiše stohastična spremenljivka poznamo samo vrednost intervala in distribucijo njihove verjetnosti. Na podlagi poznavanja intervala in verjetnostnih distribucij lahko pridemo do podatkov o srednji vrednosti dogodkov in njihovi varianci. Najbo $f(x)$ gostota verjetnosti nekega dogodka. Če je zaloga vrednosti stohastičnih spremenljivk zvezna v definicijskem intervalu govorimo o zvezni distribuciji nasprotno o diskretni. Verjetnost da dogodek stohastične spremenljivke x pade v interval x_1 do x_2 dobimo z

$$p(x_1 \leq x \leq x_2) = \int_{x_1}^{x_2} f(x)dx, \quad f(x) \geq 0$$

Velja tudi da dogodek je gotov če integral vzamemo čez vse meje

$$\int_{-\infty}^{\infty} f(x)dx = 1$$

Zelo pomembna funkcija za opis stohastične spremenljivke je kumulativna distribucijska funkcija $F(x)$. Definiramo jo z integralom v funkciji zgornje meje in pomeni verjetnost opazovanega dogodka je enaka ali manjša od x.

$$F(x) = \int_{-\infty}^{x} f(x)dx$$

Po definiciji kumulativne statistike $F(x)$ predstavlja pozitivna števila iz intervala od 0 do 1, pri tem vrednost 0 dobi za spodnjo mejo integrala (začetek intervala in 1 za zgornjo mejo intervala definicije stohastične spremenljivke). Tako verjetnost da je spremenljivka x v intervalu x_a do x_b je $F(x_a)$ - $F(x_b)$.

Pričakovana vrednost stohastične spremenljivke oziroma povprečje definiramo:

$$E(x) = \mu = \int_{-\infty}^{\infty} xf(x)dx$$

Varianco pa z

$$\sigma^2 = \int_{-\infty}^{\infty}(x-\mu)^2 f(x)dx$$

V primeru da stohastična spremenljivka zavzame diskretne vrednosti $x_i(i = 1,2,3,..,n)$ z verjetnostjo $p_i(i = 1,2,3,...n)$ in $\sum p_i = 1$ potem izraze za kumulativno, povprečje in varianco dobimo po formuli:
Kumulativna distribucija

$$F(k) = \sum_{i=1}^{k} p_k$$

povprečje ,

$$\mu = \sum_{i=1}^{kn} x_i p_i$$

in varianca

$$\sigma^2 = \sum_{i=1}^{k}(x_i - \mu)^2 p_i$$

Poleg opisanih metodologij poznamo tudi simulacijo s pomočjo metode Monte Carlo, ki temelji na ocenjevanju integralske ploščine s pomočjo naključnih generatorjev ter povprečnih vrednosti.

4.2 Generator naključnih števil

Za potrebe računalniške simulacije, ki je v svojem bistvu narave, uporabljamo generator naključnih števil. Generator naključnih števil služi za generiranje časa prihajanja in časa strežbe strank v sistem po nekem verjetnostnem zakonu. Lahko ga predstavimo kot sistem, kateri lahko zavzame $x_1, x_2, ...x_n$ stanj z verjetnostjo $p(x_1) = p(x_2) = ..p(x_n) = 1/n$. Kot primer takšnega generatorja lahko si vzamemo pravo kocko. Kocka ima šest strani in pri metu vsaka stran ima verjetnost izida 1/6. V primeru da vsaka stran kocke ima verjetnost 1/6 takšni kocki rečemo prava kocka, v nasprotnem pa lažna. Tudi kovanec je lahko generator naključnega števila vendar verjetnost izida je le 1/2. Seveda če si vzamemo 3 prava kovanca potem število neodvisnih in med sabo enako verjetnih stanje 8. Če si vzamemo dve pravi kocki izid neodvisnih stanj je 36. Še boljši generator naključnih števil, je rulet za hazardne igre, po čem se simulacija pogosto imenuje metoda Monte Carlo. Vendar, za potrebe simulacije to je premalo, poleg tega pa je še nerodno za računalniško simulacijo. Obstajajo tabele naključnih števil, katere bi bile dobri generatorji vendar bi jih bilo treba shraniti v računalniški pomnilnik poleg tega pa bi bilo nerodno izvajanje različnih simulacijskih eksperimentov. Nemogočnost ponovitve in vpogleda v eksperiment. Zaradi tega se je od vsega začetka razvoja računalnikov pristopilo razvoju numeričnih naključnih generatorjev.

4.3 Psevdo naključni generator števil

Obstaja več načinov in metod za nuerično generiranje naključnih števil. Skupno vsem postopkom je da temelje na rekurzivnimi postopki. Iz izbranega začetnega števila, ki mu rečemo seme, generiramo naslednje število. Ker od generatora naključnih števil zahtevamo da so enakomerno porazdeljeni na definicijskem področju in da so dogodki med sabo neodvisni to je očitno da z numeričnim metodama ne moremo zares da konstruiramo generator naključnih števil. Očitno gre za deterministični postopek z katerim skušamo da generiramo stohastični proces. Zaradi tega numeričnim generatorjem rečemo pseudonaključni. Navedli bomo nekaj znanih postopkov generiranja naključnih števil.

4.3.1 Generator sredine kvadrata

En od prvih generatorjev naključnih števil predložil je John von Neuman (1946 po (Žiljak)) t.i. generator sredine kvadrata. Postopek je naslednji: naj bo r_0 neko začetno število generatorja, seme. Da dobimo naslednje naključno število r_1, r_0 kvadriramo in vzamemo srednja števila. Postopek nadaljujemo vse dokler se ne pojavi rezultat enak začetnem številu ali pa je rezultat enak 0. V prvem primeru rečemo da je nastopila perioda generatorja v drugem da se je generator izpridil. Na primer če si izberemo za r_0=87 in ga kvadriramo ter vzamemo srednji dve števili dobimo naslednje naključno število r_1.

Proces nadaljujemo po rekurzivni formuli $r_i = r_{i-1}^2$, $i = 1,2,..$ kot sledi:

$$
\begin{array}{ll}
r_0 = 87 & r_0^2 = 7569 \\
r_1 = 56 & r_1^2 = 3136 \\
r_2 = 13 & r_2^2 = 0169 \\
r_3 = 16 & r_3^2 = 0256 \\
r_4 = 25 & r_4^2 = 0625 \\
r_5 = 62 & r_5^2 = 3844 \\
r_6 = 84 & r_6^2 = 7056 \\
r_7 = 05 & r_7^2 = 0025 \\
r_8 = 02 & r_8^2 = 0004 \\
r_9 = 00 &
\end{array}
$$

Izkaže se da ta generator ni enakomeren in hitro konvergira k ničli.

4.3.2 Generator sredine zmnožka

Generator je podoben prejšnjem. Definiramo ga z formulo $r_i = r_{i-1} \cdot r_{i-2}, i = 2,3,..n$. Izberimo dva začetna števila, semena, zmnožimo jih in vzamemo sredino. Postopek nadaljujemo dokler se generator ne doseže periodo ali pa se izpridi. Na primer, vzemimo za $r_0 = 87$ in $r_1 = 56$ iz prejšnjega generatorja in po formuli izračunajmo zaporedje števil :

$r_2 = 87$, ker je $r_0 r_1 = 4872$

vidimo da se je že pri tretjem številu perioda generatorja zaključila. Pa skušajmo srečo z drugim semenoma. Vzemimo za semeni drugo in tretje število iz prejšnjega generatorja $r_0 = 56$ in $r_1 = 13$ in določimo zaporedje števil $r_i =$ (56, 13, 72, 93, 69, 99, 83, 21, 74, 73, 40, 92, 68, 25, 70, 75, 40, 00). Vidimo da za druga začetna števila generator je precej daljši. Teoretično, z dvomestnim generatorjem lahko generiramo 100 naključnih števil. Podobno kot pri prejšnjem generatorju enakomernost števil je zelo slaba. Na oko se vidi dominacija višjih števil.

4.3.3 Logistična enačba kot generator naključnih števil

Zgolj iz radovednosti oglejmo si še diskretno logistično enačbo (Mk), ki jo lahko uporabimo kot generator naključnih števil ranga prejšnja dveh generatorjev. Za to uporabimo enačbo:

$$r_{i+1} = r_i c(1 - r_i)$$

Za *c=3.75* in $r_1 = 0.10$ dajemo zaporedje nekaj naključnih števil $r_i =$(0.10, 0.33, 0.83, 0.53, 0.93, 0.24, 0.68, 0.82, 0.55, 0.92, 0.28, 0.75,..). Tudi zgornja dva primera brez težav preslikamo v interval (0,1) tako da delimo z 100 (največja vrednost dvomestnih števil).

4.3.4 Kongruenčni generator

Resen razvoj pseudonaključnih generatorjev naključnih števil, ki so bili kvalitetni (neodvisnost dogodkov in enakomernost distribucije) in pogodni za računanje postavil je Lehmer 1951 (21, cf. (Žiljak)):

$$r_{i+1} = (ar_i + b)(\bmod m) \quad i = 0,1,2,3,...$$

kjer so a in b konstante r_0 začetno število oziroma seme generatorja in m veličina številke oziroma dolžina računalniške besede v računalniku. Postopek je naslednji: za skrbno izbrane konstante in semena genratora izraz v oklepaju enačbe (XX) delimo z m in ostanek opperacije oziroma reziduum vzamemo za naslednje število. Postopek nadaljujemo vse dokler se ne konča perioda generatoraja ali pa dokler se generator ne izpridi. Kvaliteta generatorja je odvisna od skrbno izbranih konstanti in semena. Obstaja več različnih variant za konstrukcijo kvalitetnega generatorja. Zgornji je takoimenovani mešovit generator. Za a=1 imamo aditivni in za b=0 imamo multiplikativni generator. Da pokažemo pomembnost izbora konstanti na dolžino generiranih zaporedij bomo pokazali nekaj primerov za multiplikativni generator.

Primeri:
Za m=10, a=7, r_0=4 imamo:

$r_1 = 8$, ker je $7 \cdot 4 = 28$, $\frac{28}{10} = 8$, ostanek je 8

$r_2 = 6$, ker je $7 \cdot 8 = 56$, $\frac{56}{10} = 5$, ostanek je 6

$r_3 = 2$, ker je $7 \cdot 6 = 42$, $\frac{42}{10} = 4$, ostanek je 2

$r_4 = 4$, ker je $7 \cdot 2 = 14$, $\frac{14}{10} = 1$, ostanek je 4

$r_5 = 8$, ker je $7 \cdot 4 = 28$, $\frac{28}{10} = 2$, ostanek je 8

Dobili smo zaporedje števil $r_i =$(4, 8, 6, 2, 4, 8,..). Z $r_5 = 8$, zaporedje se ponavlja.

Za m=7, a=5, r_0=3 imamo:
$r_i =$(3, 1, 5, 4, 6, 2, 3, 1, ..).

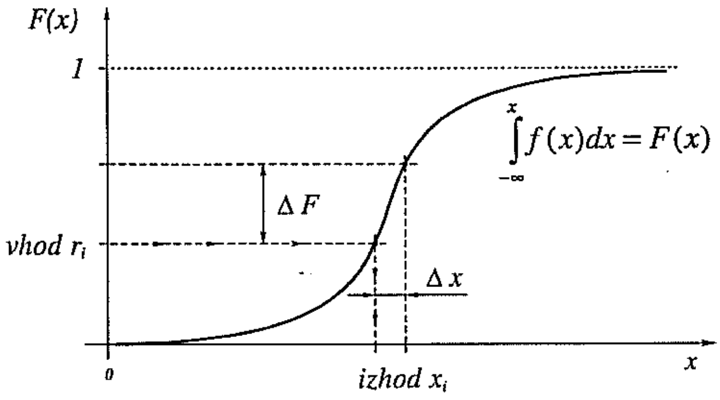
Primer za 5-bitni računalnik maksimalno število ki ga lahko zapišemo je 31, pa je m=31:

Podani sta začetni vrednosti:

a=13
r_0=7
r_1=7*13(mod 31),
r_1=9, ker je $7 \cdot 13 = 91$, $\frac{91}{31} = 2$, ostanek je 29
r_2=29*13(mod 31)
r_2=5 itn.,
Zaporedje naključnih števil pa je $r_i =$(7,29, 5,3,8,11,19,30, 18,17,4,21,25,15,9,24,2,0).
Za digitalne računalnike priporočajo se naslednje vrednosti za konstante generatorja: $m = 2^n - 1$, $a = 2^{n/2} \pm 5$, kjer je n dolžina registra računalnika. Za potrebe simulacije ponavadi rabimo naključna števila iz intervala (0,1). Normalizacijo naključnih števil dobimo tako da vsako število delimo z največjo vrednostjo generatorja tj. r_i / m.

4.4 Test naključnosti

Preden uporabimo generator naključnih števil moramo preveriti njihovo dejansko naključnost in enakomernost distribucije. Od kvalitete generatorja so odvisni rezultati simulacijskih eksperimentov. Ker so generatori pseudo jih moramo testirati da ugotovimo njihove dejanske lasnosti. Obstaja veloko različnih statističnih testov za ugotavljanje kvalitete generatorjev. Tu si bomo ogledali le nekaj osnovnih. Sicer vsak proizvajalec simulacijskih programov vključuje že istestiran generator naključnih števil.



Slika 7: Inverzna metoda

$$\lim_{\Delta x \to 0} \frac{\Delta F}{\Delta X} = f(x)$$

Primer: $f(x) = \begin{Bmatrix} 2(1-x) & 0 \le x \le 1 \\ 0 & drugje \end{Bmatrix}$

Za podano funkcijo *f(x)* poišči generator naključnih števil.

Kumulativna funkcija:

$$F(x) = \int_0^x 2(1-x)dx = 2(x - \frac{x^2}{2})$$

$$r_i = F(x) = 2(x - \frac{x^2}{2})$$

$$x_i = 1 - (1 - r_i)^{\frac{1}{2}}$$

V primeru, da je r_i naključno število po enakomerni porazdelitvi so x_i števila porazdelitve podane s funkcijo *f(x)*.

4.7.3 Eksponentna funkcija

Gostota verjetnostne porazdelitve je podana kot:

$$f(t) = \lambda e^{-\lambda t}$$

Kumulativna funkcija:

$$y = \int_{-\infty}^{t} f(t)dt = 1 - e^{-\lambda t} \quad / \ln$$

$$\lambda t = -\ln(1-y)$$

V primeru, da vstavimo namesto y naključno število r_i, dobimo:

$$t = \frac{-\ln(r_i)}{\lambda} = -T_a \ln(r_i)$$

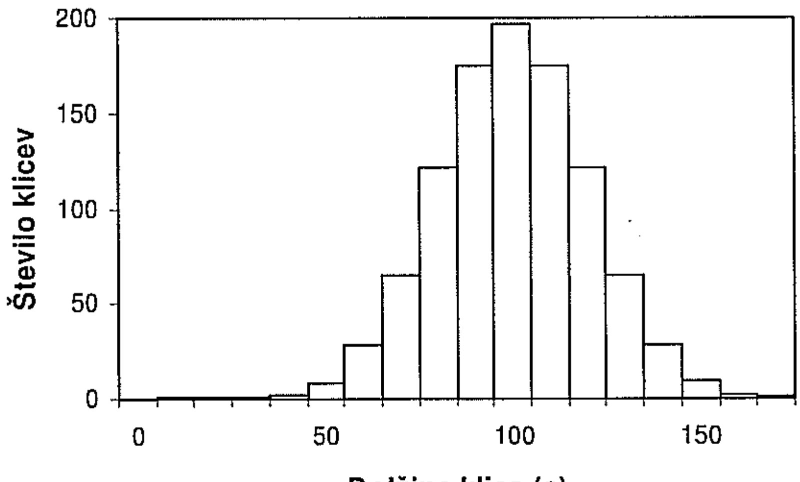
Pri tem smo upoštevali sledeče: v primeru, da je $1-r_i$ naključno število, potem je tudi r_i naključno število. Eksponentno porazdelitev števil dobimo torej tako, da logaritem z osnovo dve enakomerno porazdeljenih naključnih števil pomnožimo s povprečnim časom T_a.
Za ostale distribucije inverzna metoda ni uspešna, saj ne poznamo analitičnega izraza za kumulativno funkcijo. V takšnih primerih uporabimo numerični približek funkcije.
Poissonova porazdelitev

4.8 Numerična definicija zvezne verjetnostne funkcije.

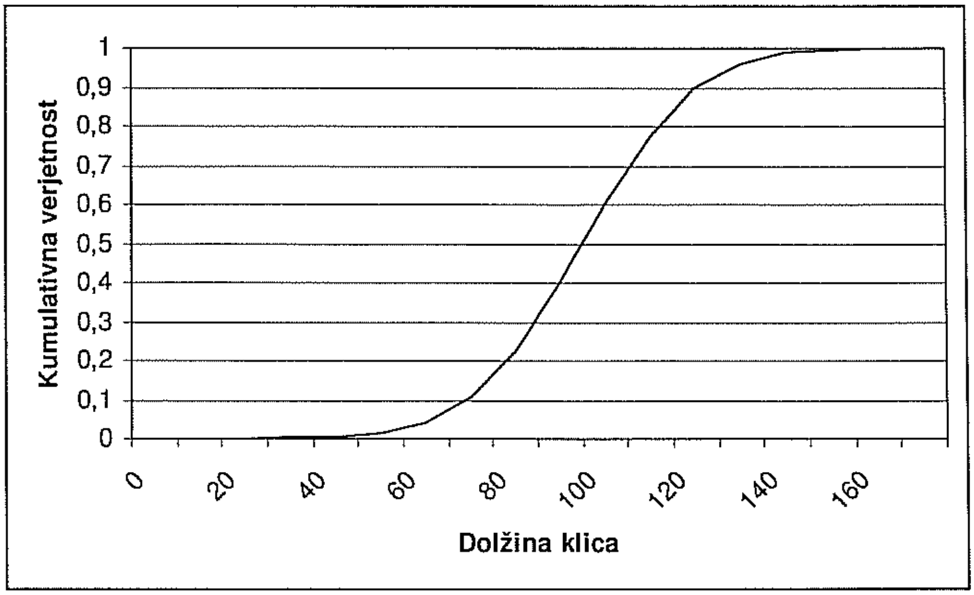
V zgoraj opisanih primerih ko iz zvezne distribucije poiščemo njeno kumulativno iz inverzno metodo poiščemo ustrezni generator naključnih števil. Toda za nekatere znane zvezne distribucije ne moremo analitično določit kumulativno funkcijo naprimer Normalna distribucija. Za nekatere zvezne procese distribucijo dobimo eksperientalno z meritvami. V takšnih primerih generator naključnih števil konkretne distribucije določimo numerično kot smo to naredili za diskretno distribucijo enačba ll.
Ponavadi merjene podatke organiziramo v tabele izbranega intervala kot frekvenčno distribucijo. Npr. (Gordon, 1969) merjenje dolžine telefonskih klicev, kot stohastično spremenljivko beležimo število klicev v interval širine 10 s. Rezutat meritev na populaciji 1000 strank prikazuje Tabela 4. Prvi stolpec predstavlja dolžino klicev, drugi pa število klicev, tretji stolpec pa relativno frekvenco, ki jo dobimo da drugi stolpec podelimo z celotno populacijo. Zadnji stolpec pa predstavlja kumulativno distribucijo, ki ga dobimo po formuli $F_r = \sum_{i=1}^{r} p_i$, ($r = 1,2,..n$) predstavlja kumulativno funkcijo stohastične spremenljivke x_r ($r = 1,2,..n$). Isti rezultat lahko pokažemo tudi grafično kot je to prikazano v nadaljevanju. Na sliki prikažemo kumulativno funkcijo zgornje distribucije tako da na abscisi nanesemo stohastično spremenljivko na ordinato pa njeno kumulativno funkcijo. Za sekvenco naključnih števil enakomerne distribucije zlahka dobimo sekvenco naključnih telefonskih klicev.

Dolžina klica (s)	Število klicev	Relativna frekvenca	Kumulativna distribucija
0	0	0.000	0.000
10	1	0.001	0.001
20	1	0.001	0.002
30	1	0.001	0.003
40	2	0.002	0.005
50	8	0.008	0.013
60	28	0.028	0.041
70	65	0.065	0.106
80	121	0.121	0.227
90	175	0.175	0.402
100	197	0.197	0.599
110	175	0.175	0.774
120	121	0.121	0.895
130	65	0.065	0.960
140	28	0.028	0.988
150	9	0.009	0.997
160	2	0.002	0.999
170	1	0.001	1.000

Tabela 4: Razporeditev dolžine telefoniranja (Gordon, 1969)



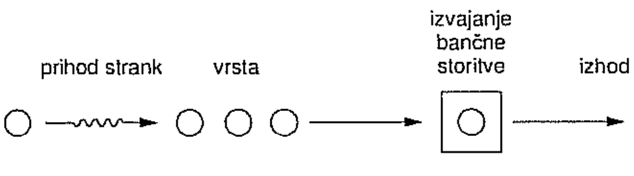
Slika 8: Razporeditev časov dolžine telefonskih klicev



Slika 9: Kumulativna funkcija distribucije

4.9 Primer dogodkovne simulacija: Banka

Za predstavitev ideje simulacije ogledali si strežbo strank v banki. Predpostavimo da je prihod strank v banko t_{ai} naključni dogodek podan z distribucijsko funkcijo f(tai). V primeru da je bančni uslužbenec prost gre stranka takoj k bančnem okencu v nasprotnem primeru, stranka počaka v vrsti. Predpostavimo, da je mehanizem strežbe »prvi pride, prvi gre« ali FIFO disciplina vrste. Čas čakanja na strežbo je odvisen od števila strank in od hitrosti opravljanja bančnih storitev. Čas strežbe strank t_{si} je naključni proces tudi podan z neko distribucijsko funkcijo g(tsi). Predpostavljamo, da bančni uslužbenec opravi istočasno le storitev za eno samo stranko oz., da ni možno, da bi izvedel bančno storitev za istočasno za dve stranki. Po bančni storitvi stranka zapusti banko. Dogodek, da stranka zapusti banko (oz. sprosti vrsto za črto zaupanja) omogoči naslednji stranki, da lahko opravi bančno storitev. S tem se je spremenila dolžina vrste v banki. Za opis sprememb stanja diskretnega sistema v določenem časovnem trenutku uporabljamo pojem dogodek. Ponavadi nas pri simulaciji strežnih sistemov zanima število ljudi v vrsti (dolžina vrste), čas čakanja na strežbo, čas strežbe, zasedenost strežnega mesta itd. Te lastnosti so odvisne od hitrosti prihoda strank v banko, časa opravljanja bančne storitve (strežbe) in vrste časovnih distribucij. Slika 10 prikazuje shemo strežbe ter gibanja strank v banki.



Slika 10 Shema strežbe v banki

Ročna simulacija:

Tabela 5 prikazuje čase prihoda t_a in čase izvajanja bančne storitve t_s z ustrezno verjetnostjo - distribucija časov

Čas med prihodom t_a	Verjetnost	Čas izvajanja bančne storitve t_s	Verjetnost
5	1/6	5	1/6
10	2/6	10	1/6
20	3/6	15	4/6

Tabela 5: Distribucija časov

Iz zgornjih podatkov je potrebno generirati čas prihoda strank in tudi čas izvajanja bančne storitve. Zato rabimo generator naključnih števil. V našem primeru, zaradi lažje ponazoritve ideje simulacije, bomo uporabili kocko. Predpostavimo da je kocka prava kar pomeni, da so verjetnosti, da pade določeno število pik med seboj enake (P(1)=P(2)=...,= P(6) = 1/6). Za ta namen bomo vrednostim, ki jih podaja **Tabela 5**, priredili ustrezne vrednosti števila pik na kocki. Rezultate prireditve podaja **Tabela 6**.

Čas prihoda strank			Čas pregleda		
Izid meta kocke	verjetnost	t_a	Izid meta kocke	verjetnost	t_s
1	1/6	5	1	1/6	5
2	1/6	10	2	1/6	10
3	1/6	10	3	1/6	15
4	1/6	20	4	1/6	15
5	1/6	20	5	1/6	15
6	1/6	20	6	1/6	15

Tabela 6: Vrednosti časa prihoda in časa strežbe v odvisnosti od števila pik na kocki

Za začetek simulacije je vse pripravljeno. Potrebno je prirediti tabelo v katero bomo beležili posamezne dogodke pomembne za oceno obnašanja sistema. Potrebno je določiti čas oziroma število obiskovalcev v sistemu strežbe.

V Tabeli 3 so prikazani časi prihoda strank ter čas strežbe. Prav tako je podana dolžina čakalne vrste ter čas čakanja v vrsti. Upoštevamo, da se delo v banki začenja ob 8:00.
Z metom kocke, ki ponazarja prihod prve stranke, katerega število pik je bilo 6 s pomočjo Tabele 6 določimo, da bo prva stranka prišla v banko čez 20 minut. To pomeni, da bo prva stranka prišla ob 8:20. Ker je bil bančni uslužbenec prost bo takoj pričel z izvajanjem bančne storitve, zaradi katere je prva stranka prišla v banko. Potrebno je opredeliti čas izvajanja bančne storitve . Z metom kocke, katerega izid je bil 6 pik določimo čas izvajanja bančne storitve 10 minut. Torej bo čas zaključka izvajanja bančne storitve ob tz= tp+ts. Iz tabele lahko razberemo da je v tem simulacijskem ciklu dolžina vrste enaka 0 (vrsta prazna), čas čakanja na bančno storitev je prav tako 0, bančni uslužbenec pa je bil prost 20 min. V drugem taktu simulacije je izid meta kocke 3, kar pomeni, da bo naslednja stranka prišla čez 10 min. Čas prihoda bo torej tp=tp+ts. Ker bo prva stranka postrežena ob času 8:30 je to tudi začetek

opravljanja storitve ob 8:30 in bo vrsta prazna, tako, da bo čas čakanja na strežbo tudi enak 0. Met kocke bo določil čas pregleda novo prispele stranke, v našem primeru 10 min, ker je bil izid meta kocke 4. Čas zaključka pregleda bo torej 8:30 + 10 =8:40.Vidimo, da strežnik v tem taktu simulacije ni bil prost. Naslednji met kocke opredeli prihod naslednje stranke ob 8:50, ker je bil izid meta kocke 5. Ker je predhodna stranka bila postrežena ob 8:40 je uslužbenec lahko pričel z opravljanjem storitve šele potem, ko je naslednja stranka prišla, torej ob 8:50. Pri tem je čakal na stranko 10 min. V času izvajanja simulacije se zaradi naključnosti prav lahko pripeti, da imamo v vrsti eno ali več strank, ki čakajo na strežbo. Proces smo zaključili ko smo generirali 11 strank.

Za t_a in t_s bomo uporabili naključni generator – kocko $P(x_1),\ldots, P(x_6) = 1/6$, kar nam prikazuje Tabela 7.

prihod			strežba		
število pik na kocki	verjetnost	t_a	število pik na kocki	verjetnost	t_s
1	1/6	5	1	1/6	5
2	1/6	10	2	1/6	10
3	1/6	10	3	1/6	15
4	1/6	20	4	1/6	15
5	1/6	20	5	1/6	15
6	1/6	20	6	1/6	15

Tabela 7: Uporaba naključnega generatorja – kocke za generiranje časa strežbe ter časov med prihodi glede na izid meta kocke

Tabela 8 prikazuje čase prihodov strank ter čas izvajanja bančne storitve oz. strežbe za posamezno stranko. Prav tako je podana dolžina čakalne vrste ter čas čakanja v vrsti. Upoštevamo, da se banka odpre ob 8:00.

število strank	naključna števila RN	t_a	t čas prihoda	pričetek strežbe	dolžina vrste	čas čakanja v vrsti t_w	naključna števila RN	t_s	zaključek strežbe t_e $t_e=t+t_s$	strežnik prost
1	2	10	8:10	8:10	0	0	3	15	8:25	10
2	3	10	8:20	8:25	1	5	4	15	8:40	0
3	5	20	8:40	8:40	0	0	5	15	8:55	0
4	5	20	9:00	9:00	0	0	4	15	9:15	5
5	6	20	9:20	9:20	0	0	6	15	9:35	5
6	1	5	9:25	9:35	1	10	3	15	9:50	0
7	2	10	9:35	9:50	1	15	1	5	9:55	0
8	5	20	9.55	9:55	0	0	5	15	10:10	0
9	2	10	10:05	10:10	1	5	2	10	10:20	0
10	5	20	10:25	10:25	0	0	4	15	10:40	5

Tabela 8:Časi prihoda in strežbe

Tabela 8 omogoča izračun nekaterih statističnih parametrov pomembnih za razumevanje delovanja sistema:

1. Povprečna izkoriščenost delovnega mesta =

$$= \frac{\text{čas zasedenosti uslužbenca}}{\text{čas simulacije}} = \frac{180-35}{180} = 0.80$$

2. Povprečni čas čakanja stranke v vrsti=

$$= \frac{\text{skupni čas čakanja stranke v vrsti}}{\text{število obiskovalcev}} = 3{,}5 \text{ min./stranki}$$

3. Povprečni čas postrežbe stranke =

$$= \frac{\text{skupni čas strežbe}}{\text{število strank}} = \frac{135}{10} = 13.5$$

4. Povprečni čas med prihodi =

$$= \frac{\text{vsota časov med prihodi}}{\text{število strank}} = \frac{145}{10} = 14.4$$

Te in veliko drugih podatkov pomembnih za analizo simuliranega problema dobimo kot standardni izpis simulacijskih jezikov.

5 GPSS

GPSS je kratica, ki pomeni **General Purpose Simulation System** ali simulacijski sistem za splošne namene. Običajno nam pomeni dvoje: programski interpreter z imenom GPSS in simulacijski jezik GPSS.

GPSS je bil razvit v začetku 60-ih let in se je ohranil in razvijal do današnjih dni. To nedvomno potrjuje njegov koncept delovanja in uporabe. Kot simulacijski jezik je GPSS procesno orientiran (dogodkovno orientirani simulacijski jeziki so npr. SIMSCRIPT, GASP). Predvsem je po našem mnenju GPSS primeren za: dinamične modele, modele z diskretnimi stanji, modele z diskretnim časom in kritičnimi dogodki ter za razne tipe stohastičnih modelov (npr. M/M/1, M/M/n, M/G/1).
Koncept GPSS
Dve entiteti opredeljujeta filozofijo GPSS. To sta BLOK in TRANSAKCIJA. Bloki so neke vrste postaje, preko katerih potujejo transakcije. Transakcije so s samo sintakso jezika pravzaprav nedosegljive; predstavljajo nekakšne elementarne delce, ki se gibajo od postaje (bloka) do postaje (bloka) po vnaprej določenem voznem redu. Vozni red je določen z diagramom poteka: zaporedje blokov in logični pogoji za preusmerjanje. Posebej pomembno je dejstvo, da GPSS daje uporabniku občutek za istočasnost več dogodkov. V simulacijskem modelu lahko predvidimo, da se istočasno dogodi več dogodkov: izdelek ali skupina izdelkov pride pred obdelovalni stroj, enega ali več izdelkov stroj obdeluje, eden ali več izdelkov lahko zapusti obdelovalni stroj,... Iluzija o sočasnosti dogajanja pa še ne pomeni, da GPSS lahko nadzira več procesov hkrati (to bi bilo mogoče samo na več procesorskem računalniku z možnostjo paralelnega procesiranja). Interpreter namreč poišče prvo transakcijo in jo pelje po predpisanih postajah. Ko se ta transakcija na neki postaji ustavi (npr. izdelek mora čakati v vrsti, da je obdelovalni stoj prost), GPSS poišče naslednjo transakcijo, ki se je časovno dogodila istočasno ali takoj za prvo. Postopek se ponavlja, dokler traja simulacijski tek. Dolžina simulacijskega teka je določena z začetnim in končnim časom ali pa s številom transakcij, ki pridejo do neke končne postaje. Simulacijsko uro neposredno krmili GPSS. Ob začetku simulacije je model prazen (npr. v obdelovalnem stroju ni nobenega izdelka, v križišču ni nobenega vozila), kasneje pa se začne polniti tako, kakor to zahtevajo parametri pojavljanja transakcij. Ob koncu simulacije GPSS sam poskrbi za izpis osnovnih statistik. Z dodatnimi stavki pa je mogoče izpisati tudi podrobnosti, ki zanimajo uporabnika.

Pomembna lastnost GPSS je možnost komunikacije s podprogrami v kakem višjem programskem jeziku (npr. FORTRAN, PASCAL, C, BASIC). To nam omogoča blok HELP. K temu se ponavadi zatečemo, če v okviru GPSS-a ne najdemo možnosti za rešitev našega problema (npr. neugodno strukturiran problem), čeprav GPSS sam nudi zelo široke možnosti:

- inicializacija konstant
- generiranje vrednosti funkcijam in spremenljivkam
- pogojno in brezpogojno razvejanje
- ponavljanje v zanki

Osnovni elementi GPSS-a

1. Transakcije
2. Resursi (viri) in druge entitete
3. Bloki

4. Krmilni stavki
5. Prevajalniški ukazi
6. Standardni numerični atributi - SNA

Transakcije so dinamične, viri pa statistične entitete v sistemu, ki ga (trenutnu) modeliramo. Bloki predstavljajo akcije prejete od transakcij. Krmilni stavki definirajo entitete,ki se rabijo v modelu ter upravljajo in vodijo izvršitev blokovnih stavkov v modelu. Prevajalniški ukazi pošljejo informacijo GPSS-h-ju,ki je potrebna za (uspešen) prevod modela. SNA so funkcije, ki zagotovijo informacijo o entitetah in izvršijo določen tip kalkulacije. Blokovni diagram opiše potek transakcije skozi sistem.

Transakcije

So bistven element v GPSS/h-ju. Kreirane so z namenom, da izvršijo določeno nalogov modelu. V večini modelih, ko transakcije konča svojo nalogo, zapustijo sistem in jih uničimo. Te transakcije so začasne entitete. V nekaterih modelih, ena ali več transakcij nikoli ne zapusi sistema. Te transakcije so permanentne.
V mnogih situacijah transakcija predstavlja osebo. Lahko pa predstavlja tudi objekt. Ima lahko enega ali več atributov. V veliko kompleksnih modelih je nekaj tipov transakcij. Vsak tip predstavlja nekaj različnega.

Resursi

So statistične, permanentne entitete v GPSS/h simulacijskem modelu. Ne morejo zapustiti sistema in ne moremo jih uničiti. Uporabljajo jih transakcije, da izvršijo določeno nalogo.Transakcija lahko zahteva več kot en resurs za izvršitev naloge.
Resursi so lahko ljudje ali objekti. V bolj kompleksnem modelu uporabimo več resursov.

GPSS/h ima dva tipa resursov:
a) Facility (pripomoček, npr. prodajalec) je namenjen za modeliranje z eno enoto resursov oz. kapaciteta mehanizma strežbe je ena,
b) storage (pomnilnik) pa za več enot resursov npr. kapaciteta mehanizma strežbe je 3.

Bloki

So namenjeni za opis premikov transakcij skozi sistem. Bloki predstavljajo specifično akcijo ali dogodek, ki (lahko) pojavi v sistemu. Kombinacija blokov predstavlja potek transakcij kot se te premikajo skozi sistem. Ponavadi rečemo temu poteku (postopku) logika simulacijskega modela.

Bloki , ki se najpogosteje uporabljajo in njihov namen:

GENERATE - kreira in vpelje transakcije v model
TERMINATE - uniči transakcije, ko te zapustijo model

ADVANCE - zadrži transakcijo za določen čas (npr. čas postopka, čas transporta)

SEIZE - zasedba mehanizma strežbe (facility)
RELEASE - sprostitev facility

ENTER - zasedba ene ali več enot meh. strežbe (storage)
LEAVE - sprostitev ene ali več enot meh. strežbe (storage)

TEST - pošlje transakcijo v nesekvenčni blok...

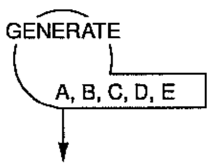
TRANSFER - pošlje transakcijov nesekvenčni blok...

Prednosti GPSS-a v primerjavi z drugimi simulacijskimi jeziki :

- lahko in hitro se ga naučimo
- ni potrebno poznati tehnike programiranja in same računalniške konfiguracije (strojne in programske opreme)
- osnovne statistike se avtomatsko izpišejo
- bolj kot pri drugih jezikih je pri GPSS poudarjena povezava med realnimi in simulacijskimi elementi
- kratki programi, močni ukazi

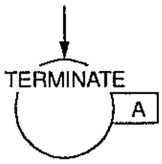
Slabosti pa nam predstavljajo naslednja dejstva:

- GPSS ni standardni jezik (po ANSI). To pomeni, da ima vsak proizvajalec prosto izbiro, kaj mu kak blok pomeni
- dela le kot interpreter, zaradi tega so ima večje zahteve po računalniških resursih (spomin, CPU)
- vsi problemi z GPSS niso rešljivi. V takih primerih je potrebno vključiti podprograme v višjih programskih jezikih ali pa najti in uporabiti drug simulacijski jezik, kar pomeni ponovno učenje in spreminjanje modela. GPSS se sestoji od okrog 50 blokov. Simboli blokov s pomočjo katerih gradimo blokovni diagram simulacijskega modela so prikazani v nadaljevanju.



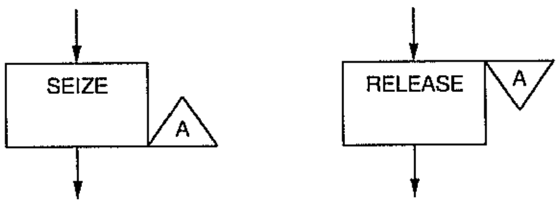
Operand	Pomen	Prednastavljena vrednost ali rezultat
A	Povprečen čas med prihodi	0.0 (nič)
B	Polovico obsega naključne spremenljivke podane z enakomerno porazdelitvijo časa med prihodi	0.0 (nič)
C	Offset interval (čas prihoda prve transakcije)	Brez prednastavljenega offset-a
D	Limitni števec (maksimalno število prihodov)	Brez prednastavljenega limitnega števca
E	Prioriteta transakcije	0 (nič)

Slika 11: Blok GENERATE z operandi A, B, C, D in E



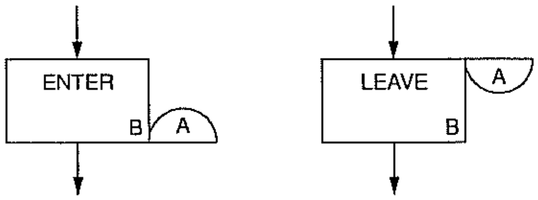
Operand	Pomen	Prednastavljena vrednost ali rezultat
A	Odštevanje števca terminacij (TC – Termination Counter) v modelu	0 (nič)

Slika 12: Blok TERMINATE z operandom A



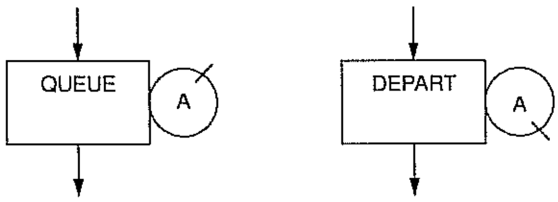
Operand	Pomen	Prednastavljena vrednost ali rezultat
A	Ime strežnega mesta tipa FACILITY ki ga zasedemo na bloku SEIZE ter sprostimo na bloku RELEASE	Prednastavljena vrednost ni podana

Slika 13: Bloka SEIZE in RELEASE z operandom A



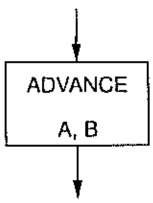
Operand	Pomen	Prednastavljena vrednost ali rezultat
A	Ime strežnega mesta tipa STORAGE (katerega eno ali več strežnih mest je zasedenih na bloku ENTER ter sproščenih na bloku LEAVE)	Napaka (error) pri prevajanju
B	Število strežnih mest, ki se zasedejo na bloku ENTER ter sprostijo na bloku LEAVE	1

Slika 14: Bloka ENTER in LEAVE z operandi A in B



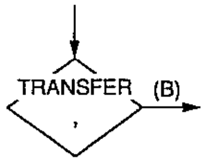
Operand	omen	Prednastavljena vrednost ali rezultat
A	Ime vrste v katero vrsta vstopa na bloku QUEUE ali izstopa na bloku DEPART	Napaka (error) pri prevajanju

Slika 15: Bloka QUEUE in DEPART z operandom A



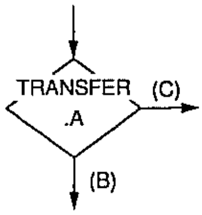
Operand	Pomen	Prednastavljena vrednost ali rezultat
A	Povprečen čas zadrževanja transakcije	0.0 (nič)
B	Polovico obsega naključne spremenljivke podane z enakomerno porazdelitvijo časa zadrževanja	0.0 (nič)

Slika 16: Blok ADVACE z operandi A in B



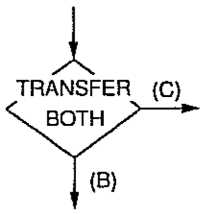
Operand	Pomen	Prednastavljena vrednost ali rezultat
A	Pri prezpogojni preusmeritvi ga moramo opustiti	
B	Kazalec na blok (kopija labele dodeljene obravnavanem bloku)	Napaka (error) pri prevajanju

Slika 17: Blok TRANSFER pri brezpogojni preusmeritvi



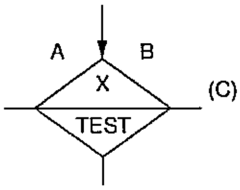
Operand	Pomen	Prednastavljena vrednost ali rezultat
A	Delež časa, pri katerem bo blok C sekvenčni oz. naslednji blok (Next Block)	Napaka (error) pri prevajanju
B	Kazalec na blok (blok B)	Blok B je sekvenčni blok
C	Kazalec na blok (blok C)	Napaka (error) pri prevajanju

Slika 18: Blok TRANSFER pri statističnem načinu preusmeritve



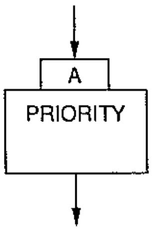
Operand	Pomen	Prednastavljena vrednost ali rezultat
A	Oba (dobeseden pomen besede BOTH)	Napaka (error) pri prevajanju
B	Kazalec na blok (blok B)	Blok B je sekvenčni blok
C	Kazalec na blok (blok C)	Napaka (error) pri prevajanju

Slika 19: Blok TRANSFER pri preusmeritvi »BOTH«



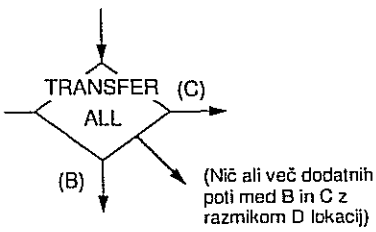
Operand	Pomen	Prednastavljena vrednost ali rezultat
A	Izraz na levi strani primerjalne operacije	Prednastavljena vrednost ni podana
B	Izraz na desni strani primerjalne operacije	Prednastavljena vrednost ni podana
C	Ime ali številka bloka na katerega je transakcija preusmerjena v primeru, da je izraz enak logični ničli (FALSE)	Prednastavljena vrednost ni podana
X (Pomožni operand)	Operacija primerjave [G], [GE], [L], [LE], [E], [NE]	Napaka (error) pri prevajanju

Slika 20: Blok TEST z operandi A, B in C ter pomožnim operatorjem X



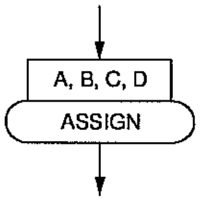
Operand	Pomen	Prednastavljena vrednost ali rezultat
A	Vrednost kot nivo prioritete, ki jo pripišemo transakciji, ki izvrši blok PRIORITY	Napaka (error) pri prevajanju

Slika 21: Blok PRIORITY z operandom A



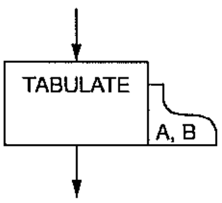
Operand	Pomen	Prednastavljena vrednost ali rezultat
A	Vse (dobeseden pomen besede ALL)	Napaka (error) pri prevajanju
B	Kazalec na blok (blok B)	Blok B je sekvenčni blok
C	Kazalec na blok (blok C)	Napaka (error) pri prevajanju
D	Celo število, ki določa razmik med bloki, ki ležijo na mestih alternativnih ubežnih rutin	Napaka (error) pri prevajanju

Slika 22: Blok TRANSFER pri preusmeritvi »ALL«



Operand	Pomen	Prednastavljena vrednost ali rezultat
A	Ime ali številka parametra oziroma niza parametrov kateremu želimo določiti neko vrednost. V primeru, da je zadnji znak pri operandu A »+« ali »−« se parametru oziroma vsem parametrom v podanem nizu poveča ali zmanjša vrednost, ki je podana s parametrom B	Prednastavljena vrednost ali rezultat
B	Vrednost, ki zamenja, se prišteje ali odšteje od parametra (parametrov), ki so podani z operandom A	Prednastavljena vrednost ni podana
C	Ime ali številka evalvacijske funkcije. Operand B je množen z vrednostjo funkcije preden se uporabi kot nadomestna vrednost, vrednost, ki jo prištevamo ali odštevamo	Prednastavljena vrednost ni podana
D	Operand D definira tip parametra. V primeru, da operand C ni uporabljen lahko stoji operand D na mestu operanda C	Prednastavljena vrednost ni podana

Slika 23: Blok ASSIGN z operandi A, B, C, in D



Operand	Pomen	Prednastavljena vrednost ali rezultat
A	Ime tabele	Prednastavljena vrednost ni podana
B	Utežni faktor	1

Slika 24: Blok TABULATE z operandoma A in B

5.1 Vhodni format GPSS/H

GPSS/H model napišemo z urejevalnikom teksta in mora biti napisan v formatu ASCII. GPSS/H prepozna tabulatorje. Bloke in ukaze pišemo z velikimi črkami, komentarje pa v poljubni obliki. Ime datoteke modela GPSS/H naj ima končnico "*.GPS".

Zaradi boljšega pregleda priporočamo naslednjo obliko zapisa programa:

```
OZNAKA OPERACIJA          OPERANDI  KOMENTAR
PRIPR  ADVANCE            3.75,1    Pripravimo sveženj
```

Komentar je ločen od operandov z najmanj enim znakom

Operande začnemo pisati v 25 stolpcu ali v predhodne stolpce

Kodo operacije začnemo pisati v 8 stolpec ali v sledeče stolpce

Oznaka (LABEL) se začne v prvem stolpcu

Polje oznake se začne v prvem stolpcu. Oznako (LABEL) predstavlja niz, ki je sestavljen iz enega do največ osem alfanumaeričnih znakov, brez šumnikov. Prvi od znakov v nizu oznake mora biti alfabetski. Ime oznake ne sme biti isto kot ime standardnega numeričnega atributa (SNA). Na splošno velja, da se to ne bo zgodilo, če bo oznaka sestavljena iz 6-8 znakov. Če pride do ponavljanja pri oznakah se nam izpiše ustrezno sporočilo o napakah. Za posamezen blok nam imena oznake ni potrebno podati. Blok potrebuje oznako, če se nanj sklicuje drug blok ali entiteta.

Za kontrolni stavek je oznaka lahko obvezna, ali pa ne. Npr. za kontrolni stavek TABLE je oznaka obvezna, za kontrolni stavek STORAGE pa oznaka ni obvezna.

Drugo polje je rezervirano za operacijsko kodo. Operacijska koda je lahko ime bloka, kontrolni stavek ali ukaz prevajalniku. V fiksnem formatu se operacijska koda začne v osmem stolpcu ali pozneje, vendar pa mora biti vedno ločena od oznake z vsaj enim praznim prostorom.

Tretje polje je namenjeno operandom. Prvi operand se imenuje operand A, drugi operand se imenuje operand B itd. Operandi so ločeni z vejicami, brez vmesnih praznih prostorov.

V četrto polje pišemo komentarje. Komentarji so namenjeni boljšemu razumevanju modela in dokumentiranju ukazov. Komentarji morajo biti ločeni od operandov z vsaj enim praznim prostorom.

Drug način za pisanje komentarjev je, da v 1. stolpec napišemo zvezdico (*), ki ji sledi naš komentar. GPSS/H teksta, ki ga napišemo v vrstico za zvezdico (*) ne upošteva.

5.2 Sintaksa GPSS

Struktura in format programa v GPSS

Sam program v GPSSu ima pravzaprav le dva dela : definicijski in proceduralni. Začetek in konec pa označujeta ukaza SIMULATE ter END.

oznak a	blok	Polja	;komentar

		A, B, C, D, E, F,	;parametri stavkov
	SIMULATE		
			;definicijski del
			;proceduralni del
	END		;konec programa

Oznaka je lahko izbirna (poljubna) glede na ime tip bloka, ki mu sledi. Lahko je simbolična (alfanumerična) ali numerična, na koncu lahko neobvezno napišemo dvopičje (:). Začenja se v prvem ali drugem stolpcu.

Blok lahko napišemo od četrtega stolpca dalje. Oznaka in blok morata biti ločena z vsaj enim presledkom.

Polja so med seboj ločena z vejico, blok in prvo polje naj bosta ločena z vsaj enim presledkom.

Komentar označimo s podpičjem, ki mora biti ločen z vsaj enim presledkom od zadnjega polja. Komentar lahko začnemo pisati tudi, če v prvi stolpec vpišemo zvezdico (*).

5.2.1 Vnos modela

Model vnašamo v obliki ASCII datoteke, ki jo pripravimo z uporabo enega izmed standardnih urejevalnikov teksta (Edit, Notepad,...). Končnica datoteke naj bo ".GPS".

5.2.2 Struktura vhodne datoteke

Vhodna datoteka je tekstovna tipa ASCII, njena blokovna sintaksa je naslednja:

<definicija simboličnih nazivov blokov>
    SIMULATE
<bloki modela>
<kontrolni stavki>
    END

Vrstni red ukaza v vhodni datoteki je naslednji:

<oznaka> <ukaz> $p_1,p_2,p_3,p_4,p_5,p_6,p_7$; komentar

kjer je:

oznaka (labela)	niz črk in številk, ki se začne s črko. Piše se od prve kolone, maksimalna dolžina je do 9 znakov.
ukaz	eden izmed kontrolnih ukazov GPSS-a ali kateri izmed blokov. Če nimajo oznake se lahko pišejo od druge kolone ali z vsaj enim presledkom od oznake naprej.
parametri	pišejo se en presledek naprej od ukaza in se ločijo z vejico. Parametri so lahko konstante, definirana simbolična imena ali kateri izmed standardnih numeričnih atributov. Nekateri parametri se v posameznih ukazih lahko izpustijo in imajo s tem privzeto vrednost. Kadar izpustimo nek parameter in za njim želimo navesti naslednjega, potem to storimo z dodajanjem vejic.
komentar	začenjamo ga z zvezdico (*) v prvi koloni ali z podpičjem (;) na katerem koli mestu v vrstici. Komentar vedno traja do konca vrstice.

6 Kontrolni ukazi

6.1.1 Osnovni stavki v GPSS

SIMULATE obvezen ukaz na začetku modela.
Stavek **SIMULATE** napišemo, kadar želimo program izvajati. Brez tega stavka interpreter samo preveri sintaktično pravilnost programa.

END obvezen ukaz na koncu modela.
Z blokom **END** program zaključimo. To je zadnji stavek v programu, obvezen in brez modifikatorjev. Ima podobno funkcijo kot END v BASICU ali FORTRANU.

START ukaz za začetek simulacije. Parametri so naslednji:
p1 - številka transakcije za katere se izvaja simulacija (glej blok TERMINATE)
p2 - je lahko NP (No Print) kadar ne želimo izpisati rezultatov simulacije.
Koliko časa traja simulacijski tek? Določimo, naj traja toliko časa, dokler ni postreženih (obdelanih) npr. 100 transakcij *(START 100).*

6.1.2 RESET in CLEAR

RESET postavi vso statistiko in vrednost relativne ure na 0, trenutnih transakcij ne odstrani iz modela, vrednost absolutne ure se ne spremeni, aktivira se relativna ura. Skupaj s stavkom START se uporablja za ugotavljanje delovanja modela v pogojih stabilnega delovanja.

CLEAR odstrani vse transakcije iz modela, absolutno in relativno uro postavi na nič, vrednost naključnega generatorja števil iz predhodnega simulacijskega teka pa ostane nespremenjena.. Uporablja se skupaj s stavkom START za izvajanje več ponovitev simulacije modela pri izvajanju simulacijskega eksperimenta.

6.1.3 Generiranje in vnos transakcij v model

GENERATE
Generator naključnih števil naj v poprečju vsakih 100 sekund generira transakcijo *(polje A = 100)*. Najmanjša razlika med dvema prihodoma je 0 sekund, največja 200 sekund *(polje B = 100)*. Časi so enakomerno porazdeljeni.

GENERATE p1,p2,p3,p4,p5,p6,p7

p1 - povprečje
p2 - odklon
p3 - čas generacije prve transakcije (offset)
p4 - maksimalno število transakcij iz bloka GENERATE (limit)
p5 - prioriteta transakcije
p6 - številka parametra transakcije (privzeto = 12)
p7 - tip parametra (H - halfword, F - fullword)

6.1.4 Umikanje transakcij iz modela

TERMINATE [p1]

umikanje transakcij iz modela, s povečevanjem števila terminiranih transakcij za ***p1***. ČE izpustimo parameter ***p1***, se število terminiranih transakcij ne menja (glej kontrolni ukaz START)

Transakcija, ki je postrežena, lahko končno zapusti sistem. Pot ene transakcije se konča z blokom **TERMINATE**, v polju A pa navedemo, da je zaključila ena (1) stranka.

6.1.5 Strežba - zadrževanje transakcij

ADVANCE
Čas strežbe traja v poprečju 80 sekund *(A=80)*, je enakomerno porazdeljen, z najmanjšim odklonom 0 sekund in največjim 160 sekund *(B=80)*. Transakcija nadaljuje (**ADVANCE**) svojo pot, ko je postrežena.

ADVANCE p1[,p2]

p1 – povprečje
p2 – odklon

6.1.6 Strežnik (FACILITY)

SEIZE, RELEASE

Z blokom **SEIZE** povemo, da se je transakcija prebila do strežnega mehanizma (strežni mehanizem je v tem trenutku prost in ga lahko transakcija pridobi).
Časovna točka, v kateri je transakcija sprostila (**RELEASE**) strežni mehanizem *STREZI* je pomembna za naslednjo transakcijo. Ta bo lahko znova zaposlila strežni mehanizem *STREZI*.
Zasedanje strežnika

SEIZE p1

p1 - ime, naziv strežnika

6.1.7 Sproščanje strežnika

RELEASE p1

p1 - ime, naziv strežnika

6.1.8 Vrste (**QUEUE**)

QUEUE, DEPART

Transakcija se mora postaviti v vrsto, ki jo imenujmo *VRSTA*. Blok, ki omogoča tvorbo vrst je **QUEUE**. Ko transakcija enkrat vstopi v vrsto, jo mora tudi zapustiti. To se zgodi v trenutku, ko transakcija pridobi strežni mehanizem. V GPSS moramo odhod iz vrste označiti z dvema blokoma. **SEIZE** pove, da se je transakcija prebila do mehanizma strežbe, ki ga imenujemo *STREZI* ter tudi fizično zapustila (**DEPART**) vrsto *VRSTA*.

6.1.9 Vstopanje v vrsto

QUEUE p1[,p2]

p1 - ime, naziv vrste
p2 – število transakcij (privzeta vrednost 1)

6.1.10 Izstopanje iz vrste

DEPART p1[,p2]

p1 - ime, naziv vrste
p2 - število transakcij (privzeta vrednost 1)

6.1.11 Kapaciteta (**STORAGE**) / Zasedanje kapacitete

ENTER, LEAVE, STORAGE

Kadar imamo več kot eno enakovredno strežno mesto uporabimo za pridobitev mehanizma strežbe blok **ENTER**, ko pa želimo, da transakcija zapusti mehanizem strežbe to storimo z blokom **LEAVE**. Ob tem moramo povedati tudi število strežnih mehanizmov z stavkom **STORAGE** *(STREZI STORAGE 2)*.

ENTER ime,[p1]

transakcija v storage ***ime*** vnese ***p1*** enot. Kadar izpustimo ***p1***, potem je ***p1=1***.

6.1.12

6.1.13 Sproščanje kapacitete

LEAVE ime,[p1]

transakcija iz bloka storage-a ***ime*** sprosti ***p1*** enot. Paziti je potrebno, da ne sprostimo več enot, kot je trenutna vsebina kapacitete.

Definiranje kapacitete

ime STORAGE kapaciteta

v storage-u ***ime*** je lahko največ ***kapaciteta*** enot.

6.1.14 Usmerjanje transakcij

Pogojno in brezpogojno usmerjanje transakcij

TRANSFER p1,p2,p3,p4

Blok usmerja transakcije na različne načine. Način je odvisen od vrednosti prvega parametra:

p1 = interval

- število transakcij ***p1*100%*** se usmerja na blok ***p3***, ostale na blok ***p2***. Parametra ***p4*** ne uporabljamo;

p1 = BOTH

- transakcija poizkuša hkrati vstopiti v blok ***p2*** in ***p3*** dokler eden ni prost. Parametra ***p4*** ne uporabljamo;

p1 = ALL

- transakcija pregleduje bloke ***p2+p4***, ***p2+2*p4***,... do ***p3*** in vstopi v prvi prost blok

p1 = PICK

- transakcija poskuša vstopiti v bloke ***p2***, ***p2+1***,... do ***p3***;

p1 = FN

- transakcija gre v blok, katerega vrednost se dodeli tako, da se vzame vrednost funkcije, katere število ali simbolično ime kaže na ***p2*** ali ***p3***;

Kadar izpustimo parameter ***p1***, gre transakcija brezpogojno na ***p2***, parametra ***p3*** in ***p4*** pa ne uporabljamo (**TRANSFER ,p2**).

6.1.15 Tabele

TABULATE ime

Z vstopom v ta blok transakcija ažurira tabelo ime

Določanje tabel

Bloku TABULATE je pridružen ukaz, ki določa vrsto in obliko tabele:

ime TABLE p1,p2,p3,p4

p1 = ime spremenljivke, ki jo tabeliramo (npr. numerični atribut M$1
Posebna primera:
p1 = *IA* (inter arrival) - tabelira se interval med dvema vstopoma transakcij v blok *TABULATE*
p1 = *MP$n* - tabeliramo razliko trenutnega stanja simulacijske ure in velikost parametra n, transakcije ki se navede z blokom *MARK*, kadar se tabelira prehodni čas.
p2 = spodnja meja (razred) tabele
p3 = širina intervala (razreda)
p4 = število intervalov

Dodatna tabela za vrste

TRANSFER
Blok TRANSFER preusmerja del transakcij na druge dele programa.
Sintaksa: *TRANSFER xx,smer1,smer2*
verjetnost *xx* velja za *smer2.*

TRANSFER p1,p2,p3,p4
transakcija z vstopom v ta blok ažurira dodatno tabelo v statistiki za vrsto *p1*. Spodnja meja je *p2*, širina intervala je *p3*, število intervalov pa je *p4.*

Standardni numerični atributi (SNA)

V tabeli 1 so navedeni nekateri standardni numerični atributi (SNA) jezika GPSS:

Domena	SNA	Pomen atributa
SIMULACIJA	c$1	trenutna vrednost simulacijske ure
TRANSAKCIJA	p$*n*	vrednost ***n***-tega parametra transakcije
	m$1	čas trajanja transakcije
	pr$1	prioriteta transakcije
BLOK	n$*b*	skupno število transakcij, ki so vstopile v blok ***b***
	w$*b*	trenutno število transakcij v bloku ***b***
STREŽNIK (FACILITY)	fc$*n*	število vstopov v strežnik (facility) ***n***
	ft$*n*	povprečni čas zadrževanja transakcije v strežniku
	fr$*n*	izkoriščenost strežnika ***n***
KAPACITETA (STORAGE)	s$*n*	trenutno število transakcij v kapaciteti ***n***
	r$*n*	število praznih mest v kapaciteti ***n***
	sc$*n*	število vstopov v kapaciteto ***n***
	st$*n*	povprečni čas zadrževanja v kapaciteti ***n***
VRSTA (QUEUE)	q$*n*	trenutna dolžina vrste ***n***
	qt$*n*	povprečni čas čakanja v vrsti ***n***
	qc$*n*	število vstopov v vrsto ***n***
	qm$*n*	maksimalna dolžina vrste ***n***
	qz$*j*	število direktnih prehodov skozi vrsto ***n***
PARAMETRI MODELA (SAVEVALUE)	x$*n*	vrednost parametra ***n*** - celo število (32 bit)
	xh$*n*	vrednost parametra ***n*** - celo število (16 bit)
	xl$*n*	vrednost parametra ***n*** - realno število
FUNKCIJE IN SPREMENLJIVKE	rn$*i*	*i*-ti generator naključnih števil (*i = 1,... 8*)
	fn$*n*	izračunana vrednost funkcije ***n***
	v$*n*	vrednost spremenljivke ***n***

tabela 9: standardni numerični atributi (SNA) GPSS-a

6.1.16 Posebne veličine posameznih transakcij

ASSIGN p1,p2

Z vstopom v ta blok transakcija v parameter ***p1*** postavi vrednost ***p2***

PRIORITY p1

Postavljanje prioritete transakcije na ***p1***. Dovoljene so vrednosti od 0 do 127

INCREMENT p1,p2

Parameter ***p1*** povečamo za ***p2***

DECREMENT p1,p2

Parameter ***p1*** pomanjšamo za ***p2***

INDEX p1,p2

Parametru *p1* prištejemo *p2* in vrednost postavimo v prvi parameter

Ukazi za definiranje logičnih izrazov in funkcij

ime FUNCTION p1,tn

x1, y1/x2, y2/.../xn,yn

ime - naziv funkcije

p1 - numerični atribut, ki definira vrednost neodvisne spremenljivke x

t - tip funkcije, ki je lahko:

- *C* - *continious*; zvezna funkcija
- *D* - *diskrete*; diskretna funkcija
- *L* – diskretna, pri čemer ima x vrednosti 1, 2, 3, 4, ... ,n
- *E* – diskretna, pri čemer so vrednosti y podane s SNA
- *M* – kombinacija L in E

n - število parov posameznih točk, ki jih ločimo z /. Vrednosti *x* morajo biti rastoče

Kopije transakcij in vzporedne poti

Razstavljanje dinamičnih objektov nastane v bloku **SPLIT**, ponovno združevanje pa v bloku **ASSEMBLE**, iz katerega pot nadaljuje samo ena transakcija.

SPLIT p1,p2

transakcijo kopiramo v *p1* primerkov in jih usmerimo v blok *p2*. Originalna transakcija nadaljuje pot v naslednji blok

GATHER p1

V tem bloku se transakcije zadržujejo dokler se jih ne nabere *p1*. Zatem nadaljuje pot skupaj

ASSEMBLE p1

V tem bloku se zbira transakcija *p1* in iz njega nadaljuje pot samo ena. Ostale transakcije so uničene

Relacije med numeričnimi atributi

TEST

Blok **TEST** dovoljuje in prepoveduje nadaljevanje pot transakcije preko bloka.

Sintaksa: *TEST_xx* *vrednost1, vrednost2, izhod*

Če je rezultat primerjave **TRUE**, transakcija nadaljuje pot skozi **TEST**, sicer se preusmeri na blok z labelo *izhod*. Vrednosti argumenta (xx) so:

L, LE, G, GE, E, NE (polje A manjše, večje, enako,... polju B).

Blok **TEST_pogoj p1, p2[,p3]** primerja dva numerična atributa ***p1*** in ***p2***. Če je relacijski argument resničen, transakcija nadaljuje pot v naslednji blok, če pa ni, odide na blok ***p3***. Pomembno je opozoriti, da naslednje transakcije, za katere je pogoj izpolnjen, vstopijo v ta blok neodvisno od tistih, za katere pogoj ni izpolnjen. Če izpustimo parameter ***p3***, transakcija čaka v predhodnem bloku dokler pogoj ni izpolnjen. Dovoljeno je testiranje naslednjih relacij:

TEST_L p1,p2[,p3]	če je ***p1*** manjše od p2
TEST_LE p1,p2[,p3]	če je ***p1*** manjše ali enako od p2
TEST_E p1,p2[,p3]	če je ***p1*** enako od p2
TEST_NE p1,p2[,p3]	če je ***p1*** različno od p2
TEST_G p1,p2[,p3]	če je ***p1*** večje od p2
TEST_GE p1,p2[,p3]	če je ***p1*** večje ali enako od p2

6.1.17 Logični pogoji

Blok **GATE_pogoj p1[,p2]** je previden za preverjanje statusa strežnega mesta (strežnika), kapacitete ali logičnega stikala. Če je pogoj, postavljen na ***p1*** izpolnjen, transakcija nadaljuje pot skozi ta blok, drugače gre na blok p2. Če izpustimo ***p2***, transakcija čaka v bloku **GATE**, dokler se pogoj ne izpolni. Dovoljeno je testiranje naslednjih pogojev:

GATE_NU p1[,p2]	transakcija nadaljuje, če je ***p1*** prost
GATE_U p1[,p2]	transakcija nadaljuje, če je ***p1*** zaseden
GATE_I p1[,p2]	transakcija nadaljuje, če je ***p1*** prekinjen pri strežbi neke transakcije (**PREEMPT**)
GATE_NI p1[,p2]	inverzno od *GATE_I*
GATE_SE p1[,p2]	transakcija nadaljuje, če je kapaciteta ***p1*** prazna
GATE_SNE p1[,p2]	transakcija nadaljuje, če kapaciteta ***p1*** ni prazna
GATE_SF p1[,p2]	transakcija nadaljuje, če je kapaciteta ***p1*** polna
GATE_SNF p1[,p2]	transakcija nadaljuje, če kapaciteta ***p1*** ni polna
GATE_LS p1[,p2]	transakcija nadaljuje, če je logično stikalo ***p1*** vklopljeno
GATE_LR p1[,p2]	transakcija nadaljuje, če je logično stikalo ***p1*** ni vklopljeno

6.1.18 Vgrajene funkcije

V GPSS so vgrajene naslednje funkcije:

RVEXPO (3,1)
RVTRI(3,0,7.5,10.0)
RVNORM(5,100,10)
ABS()
in ostale

6.2 *Primer modela GPSS/H:*

```
SIMULATE
*
*       GPSS/H Blokovni del
*
        GENERATE        4,1             Prihod svežnja na 4+-1 minute
        ADVANCE         2               Transportni čas do obdelave
        SEIZE           STREZNIK        Zasedemo linijski strežnik
        ADVANCE         3.75,1          Pripravimo svežen j
        RELEASE         STREZNIK        Sprostimo linijski strežnik
        TERMINATE       1
*
*       GPSS/H Kontrolni stavki
*
        START           100             Končam po 100 svežnjih
        END
```

6.2.1 Simulacija modela GPSS/H

V mapi, kjer se nahaja program GPSS/H vtipkamo:

GPSSH

in pritisnemo tipko <ENTER>. Izpiše se naslednje sporočilo:

ENTER SOURCE FILE NAME:

GPSSH zahteva, da vpišemo ime modela, ki ga želimo pognati. Npr. da želimo simulirati model brivskega salona, ki je shranjen v datoteki "JOEBARB.GPS". Vtipkamo:

JOEBARB

in pritisnemo <ENTER> . Končnice GPS nam ni potrebno vtipkati.
Model lahko poženemo tudi na krajši način tako, da vtipkamo:

GPSSH JOEBARB

Izpiše se nam naslednje sporočilo o delu prevajalnika in izvajanju programa:

GPSS/H MS-DOS STUDENT/DEMO RELEASE 2.0 (UG289)
FILE: JOEBARB.GPS
Compilation begins.
Pass 1 (with source listing) ...
Pass 2 ...
Simulation begins.

Sporočilo *Pass 1 (with source listing)* ... nam pove, da bo GPSS/H prebral izvorno datoteko, preveril pravilnost kode in naredil listing izvorne kode. V tej prvi fazi GPSS/H določi tudi numerične vrednosti simbolom, (npr. oštevilčenje stavkov, imen kapacitet-facility) ki se uporabljajo v modelu.

Sporočilo *Pass 2 ...* nam pove, da bo GPSS/H prevedel model v obliko, ki je primerna za izvajanje. Če v našem modelu ni nobene napake se pojavi naslednje sporočilo: *Simulation begins.* To pomeni, da se simulacija normalno izvaja. Če se med izvajanjem simulacije pojavi napaka, se nam izpiše opozorilno sporočilo na monitorju in prav tako v standardno izhodno poročilo. Standardno izhodno poročilo se zapiše v datoteko z imenom, ki je isto kot ime izvorne datoteke, le da je njegova končnica "*.LIS" (v našem primeru JOEBARB.LIS).

6.2.2 Standardna izhodna datoteka

Standardna izhodna datoteka nam v prvem delu prikaže povzetek modela, v drugem delu pa so zapisani rezultati simuliranega modela.

```
STUDENT GPSS/H RELEASE 3.0c-C10 (EP195)      19 Dec 1995    12:08:30      FILE: BRIVEC.GPS

      LINE# STMT#  IF DO  BLOCK#  *LOC       OPERATION        A,B,C,D,E,F,G    COMMENTS

          1    1                             SIMULATE
          2    2                    *
          3    3                    *        GPSS/H Blokovni del
          4    4                    *
          5    5            1                GENERATE         4,1              Prihod na 4+-1 minute
          6    6            2                ADVANCE          2                Transportni čas do obdelave
          7    7            3                SEIZE            STREZNIK         Zasedemo strežnik
          8    8            4                ADVANCE          3.75,1           Strežba
          9    9            5                RELEASE          STREZNIK         Sprostimo strežnik
         10   10            6                TERMINATE        1
         11   11                    *
         12   12                    *        GPSS/H Kontrolni stavki
         13   13                    *
         14   14                             START            100              Končam po 100 strankah
         15   15                             END

ENTITY DICTIONARY (IN ASCENDING ORDER BY ENTITY NUMBER; "*" => VALUE CONFLICT.)

           Facilities: 1=STREZNIK

SYMBOL     VALUE    EQU DEFNS  CONTEXT       REFERENCES BY STATEMENT NUMBER

STREZNIK     1                 Facility          7    9
```

V prvi vrsti na vsaki strani je zapisana verzija programa, datum in čas izvajanja simulacije. Izpis nam prikaže številko vrstice in bloka, ki nam jo poda GPSS/H v času prevajanja.

Entity dictionary oz. slovar entitet nam izpiše simbole, ki smo jih uporabili v modelu. Simbol je ime, ki je povezano z entiteto (t.j. ime bloka, ali ime enote). Ta del nam prikaže številko entitete, ki jo GPSS/H določi v času prevajanja. Npr. slovar entitet ki je prikazan v zgornjem izpisu nam pove, da pripada simbolu STREZNIK vrednost 1, kar pomeni, da sta enota STREZNIK in enota 1 isto.

Naslednji del standardne izhodne datoteke nam pove, na kaj se nanaša določen simbol (referenčni opis), ki je naveden v slovarju entitet. V našem primeru se simbol SERVER nanaša na enoto (proizvodno enoto - facility), kar je napisano v stolpcu CONTEXT. SERVER je bil uporabljen dvakrat in sicer v stavkih št. 7 in 9. Če pogledamo v izpis, sta ta dva stavka:

SEIZE   SERVER
in
RELEASE   SERVER

Slovar entitet in referenčni opis nam pride prav pri večjih modelih, zlasti pri odkrivanju in odpravljanju napak. Če zaradi večje preglednosti pri velikih modelih ne želimo, da se slovar

entitet in referenčni opis izpišeta, ob klicu GPSS/H napišemo: NODICT - slovar entitet se ne izpiše oz. NOXREF - referenčni opis se ne izpiše.
Primer:

GPSSH JOEBARB NODICT NOXREF

Naslednji izpis prikazuje rezultate simulacije. Rezultate simulacije lahko razdelimo na štiri ločene dele, ki nam posredujejo podatke o:

- uri
- blokih
- entitetah
- generatorju naključnih števil.

```
RELATIVE CLOCK: 399.2730   ABSOLUTE CLOCK: 399.2730

BLOCK CURRENT     TOTAL
1                  101
2          1       101
3                  100
4                  100
5                  100
6                  100

--AVG-UTIL-DURING--
FACILITY  TOTAL  AVAIL  UNAVL    ENTRIES    AVERAGE    CURRENT  PERCENT  SEIZING  PREEMPTING
          TIME   TIME   TIME                TIME/XACT  STATUS   AVAIL    XACT     XACT
STREZNIK  0.956                   100        3.818     AVAIL

RANDOM   ANTITHETIC    INITIAL    CURRENT     SAMPLE   CHI-SQUARE
 STREAM     VARIATES   POSITION   POSITION     COUNT   UNIFORMITY
    1          OFF      100000     100202       202      0.70
```

Izpis podatkov o uri nam pove, kakšen je bil absolutni in relativni čas ob koncu simulacijskega teka. V našem primeru sta absolutni in relativni čas enaka - 399.2730. Če privzamemo, da smo za enoto v modelu vzeli eno minuto je čas simulacijskega teka trajal 399.2730 minut.

Relativni čas simulacije nam poda čas simulacije oz. čas ki je potekel od zadnjega klica kontrolnega stavka RESET ali CLEAR (ali BRESET, BCLEAR). V modelih. ki omenjenih kontrolnih stavkov ne uporabljajo bosta absolutni in relativni čas simulacije enaka. Najmanjša vrednost relativnega časa simulacije je 0.

Absolutni čas meri celoten simulacijski čas, ki je pretekel od začetka zadnjega simulacijskega teka ali zadnjega izvajanja kontrolnega stavka CLEAR ali bloka BCLEAR. Najmanjša vrednost absolutnega časa je 0.

Podatki o blokih so razvrščeni v stolpce.
V stolpcu z imenom BLOCK se izpišejo imena blokov in številke blokov.
Stolpec z imenom CURRENT nam pove, koliko transakcij je trenutno v posameznem bloku.
V stolpcu z imenom TOTAL je izpisano število vstopov transakcij v posamezne bloke v času izvajanja simulacije.

Npr.: na koncu simulacijskega teka ob času 399.2730 je številka v stolpcu CURRENT za blok 2 enaka 1, kar pomeni, da je ob tem času transakcija v bloku 2. Številka pod stolpcem TOTAL je 101, kar nam pove, da se je do konca simulacije blok 2 izvedel 101-krat.

Podatki o entitetah so razvrščeni v stolpce in pomenijo naslednje:

- Stolpec FACILITY -> podatki o imenu ali številki enote (facility).
- Stolpec --AVG-UTIL-TOTAL-DURING-- TIME -> podatki o deležu celotnega časa, v katerem je bila enota (facility) zasedena.
- Stolpca --AVG-UTIL-TOTAL-DURING-- AVAIL TIME in UNAVL TIME -> Enota je lahko razpoložljiva ali nerazpoložljiva (v primeru popravila, malice itd.). Delež časa, v katerem je enota zasedena, vendar v stanju razpoložljivosti ali nerazpoložljivosti prikazujeta omenjena dva stolpca. Če je bila enota ves čas razpoložljiva sta ta dva stolpca prazna. Razpoložljivost določamo z blokoma FAVAIL, in FUNAVAIL. (facililty available/unavailable).
- Stolpec ENTRIES -> skupno število primerov, ko je bila enota (facility) zasedena.
- Stolpec AVERAGE TIME/XACT -> povprečen čas zasedbe enote (facility).
- Stolpec CURRENT STATUS -> podatki o tem, ali je enota razpoložljiva v danem času (available) ali nerazpoložljiva (unavailable). Razpoložljivost (available-AVAIL) je neodvisna od zasedenosti ali nezasedenosti enote (busy/idle).
- Stolpec PERCENT AVAIL -> delež celotnega časa, v katerem je bila enota razpoložljiva (prazen prostor pomeni, da je bila enota razpoložljiva 100%)
- Stolpec SEIZING XACT -> podatki o številki transakcije, ki je trenutno v enoti (facility). Če v enoti ni transakcije, je ta del prazen.
- Stolpec PREEMPTING XACTS -> število transakcij, ki so vstopile v enoto (facility) s prioriteto.

Podatki o nizu naključnih števil nam povejo, da smo uporabili samo en niz naključnih podatkov (stolpec RANDOM STREAM). Za generiranje 100 svežnjev blaga smo uporabili 202 naključni števili (stolpec SAMPLE COUNT). Ostali podatki govorijo o načinu generiranja naključnih števil (ANTHITETIC VARIATES - nasprotna varianca - izbiranje naključnih števil po principu nasprotne variance, INITIAL POSITION - zaporedna številka naključnega vzorca prvega od generiranih naključnih števil - (100000-ega), FINAL POSITION - številka zadnjega vzetega naključnega vzorca - ni nujno, da naključna števila jemljemo enega za drugim, CHI-SQUARE UNIFORMITY - delež naključnih števil, ki odstopajo od popolnega (enotno porazdeljenega) vzorca naključnih števil.

7 GPSS primeri

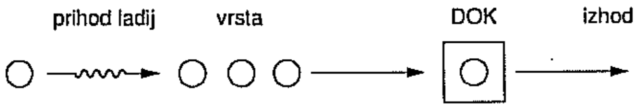
7.1 Preprosto pristanišče (luka)

Definicija modela

Tovorne ladje prispejo na razkladanje v pristanišče vsakih 16±5 ur. Pristanišče ima en razkladalni prostor (dok). V primeru, da je dok prost, ladja odpluje na razkladanje, v nasprotnem primeru pa čaka v sidrišču.

Čas razkladanja t_s = 15±10 ur. Ladja zapusti luko po končanem razkladanju.

Slika 25 prikazuje shemo prihoda ladij v luko na razkladanje v dok.



Slika 25: Shema prihoda ladij v pristanišče

Problem

Zaradi dolgega časa čakanja na razkladanje in prevelikega števila ladji na sidrišču smo opazili, da gre določeno število ladij k konkurenci.

Naloga

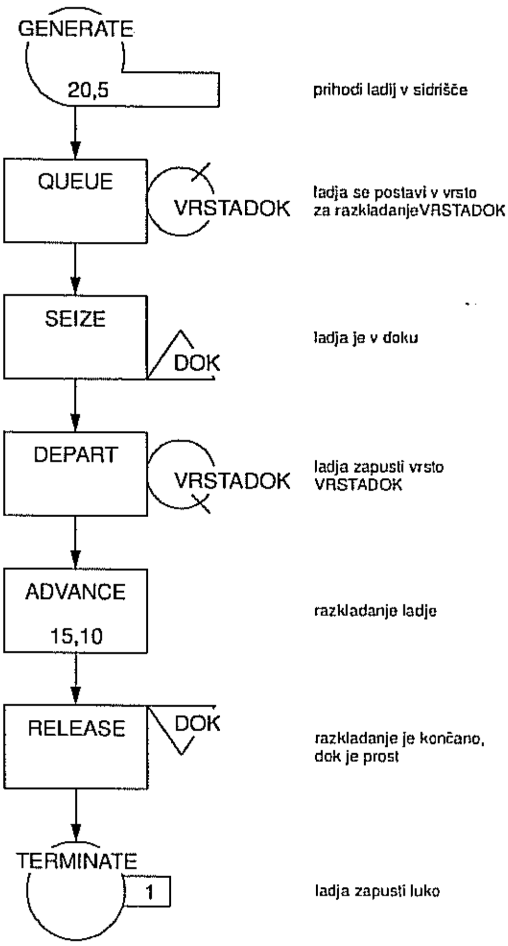
Določi čas čakanja ladij v pristanišču, čas čakanja v sidrišču in število ladij, ki čakajo na razkladanje.

Kriteriji

- maksimalno število ladij v sidrišču: 5 + 10%
- maksimalen čas čakanja: 6 dni + 10%

Blokovni diagram

Proces raztovarjanja ladij v ladijskem doku z bloki GPSS prikazuje Slika 26.



Slika 26: Blokovni diagram preprostega pristanišča

Simulacijski model

Slika 27 prikazuje simulacijski model za primer luke v simulacijskem jeziku GPSS/H.

```
         SIMULATE
*
*        Simulacija luke
*
*
*         GPSS/H blokovni del
*
*
        GENERATE     16,5             Prihod ladij v luko 20 +- 5 ur
        QUEUE        VRSTADOK         Ladja se postavi v vrsto
        SEIZE        DOK              Ladja je v doku
        DEPART       VRSTADOK         Ladja zapusti vrsto
        ADVANCE      15,10            Čas razkladanja
        RELEASE      DOK              Ladja zapusti dok - dok je prost
        TERMINATE    1                Ladja zapusti luko
*
*       GPSS/H kontrolni stavki
*
        START        100              (terminations to go) postavimo na 100
        END
```

Slika 27: Simulacijski model za primer luke

Rezultati

Slika 28 prikazuje del rezultatov simulacijskega teka za primer modela luke.

```
RELATIVE CLOCK: 1616.6732   ABSOLUTE CLOCK: 1616.6732


BLOCK CURRENT      TOTAL
1                   104
2          4        104
3                   100
4                   100
5                   100
6                   100
7                   100


           --AVG-UTIL-DURING--
FACILITY   TOTAL  AVAIL  UNAVL      ENTRIES     AVERAGE   CURRENT  PERCENT  SEIZING  PREEMPTING
           TIME   TIME   TIME                   TIME/XACT  STATUS   AVAIL    XACT      XACT
     DOK   0.984                      100        15.906    AVAIL


   QUEUE    MAXIMUM     AVERAGE     TOTAL      ZERO     PERCENT      AVERAGE     $AVERAGE    QTABLE
CURRENT
            CONTENTS    CONTENTS    ENTRIES    ENTRIES    ZEROS     TIME/UNIT    TIME/UNIT    NUMBER
CONTENTS
VRSTADOK         6        2.224       104        6         5.8       34.567       36.683
4
```

Slika 28: Delni rezultati simulacije modela luke

Iz rezultatov je razvidno, da je bil skupni čas simulacije raztovarjanja sto ladij (RELATIVE in ABSOLUTE CLOCK) 1616 časovnih enot, ki so v našem primeru ure. V nadaljevanju vidimo število ladij oziroma transakcij, ki so potovale skozi posamezne bloke modela. Tako vidimo, da je blok 1 (GENERATE 16,5) uvedel 104 transakcije, 100 transakcij pa je model zapustilo, zato se je simulacija ustavila (ukaz START 100 pomeni, da bo simulacija končana v trenutku, ko stota transakcija vstopi v blok TERMINATE). V delu rezultatov z oznako -AVG-UTIL-DURING- je razvidno, da imamo strežno mesto tipa FACILITY z imenom DOK, ki je zaseden približno 98% časa (TOTAL TIME – 0.984), da je v strežbo prišlo 100 ladij (ENTRIES – 100). Povprečni čas razkladanja ladje je bil slabih 16 ur (AVERAGE TIME/XACT – 15.906), kar ni povsem enako zahtevanemu povprečju petnajstih ur in je posledica odmika od povprečja za čas ±5 ur. V spodnjem delu rezultatov je statistika vrste, ki ima v našem primeru ime VRSTADOK V njej je bilo največ šest ladij (MAXIMUM CONTENTS), skupaj je v vrsto pripeljalo 104 ladij (TOTAL ENTRIES), brez čakanja pa jih je na dok odpeljalo 6 (ZERO ENTRIES), kar je 5,8% od vseh 104 ladij (PERCENT ZEROS), povprečni čas čakanja vseh ladij je bil 34,567 ure (AVERAGE TIME/UNIT), povprečni čas čakanja ladij, ki so dejansko čakale v vrsti pa je 36,683 ure ($AVERAGE TIME/UNIT). Ob času zaključka simulacije so v vrsti še štiri ladje, ki čakajo na raztovarjanje (CURRENT CONTENTS).

Glede na podane kriterije vidimo, da naš model zadosti drugemu kriteriju, saj je čas čakanja manjši od šestih dni, prvemu kriteriju glede na maksimalno število ladij pa ne zadosti, saj je maksimalno število ladij šest.

7.2 Banka 1

Definicija modela

Primer banke ki smo ga obravnavali z ročno simulacijo v poglavju 3 bomo v nadaljevanju simulirali z uporabo simulacijskega jezika GPSS/H.

Predpostavimo da je prihod strank v banko naključni dogodek podan z distribucijsko funkcijo:

```
PRIH    FUNCTION RN(1),D4
0,0/.16,5/.49,10/1,20.
```

V primeru da je bančni uslužbenec prost gre stranka takoj k bančnem okencu v nasprotnem primeru, stranka počaka v vrsti. Predpostavimo, da je mehanizem strežbe »prvi pride, prvi gre« ali FIFO disciplina vrste. Čas čakanja na strežbo je odvisen od števila strank in od hitrosti opravljanja bančnih storitev. Čas strežbe strank je naključni proces tudi podan z distribucijsko funkcijo:

```
STORI   FUNCTION RN(2),D4
0,0/.16,10/.33,5/1,15.
```

Predpostavljamo, da bančni uslužbenec lahko istočasno opravi le storitev za eno samo stranko oz., da ni možno, da bi izvedel bančno storitev istočasno za dve stranki. Po bančni storitvi stranka zapusti banko. Dogodek, da stranka zapusti banko (oz. sprosti vrsto za črto zaupanja) omogoči naslednji stranki, da lahko opravi bančno storitev. S tem se je spremenila dolžina vrste v banki. Slika 10 prikazuje shemo strežbe ter gibanja strank v banki.

Problem

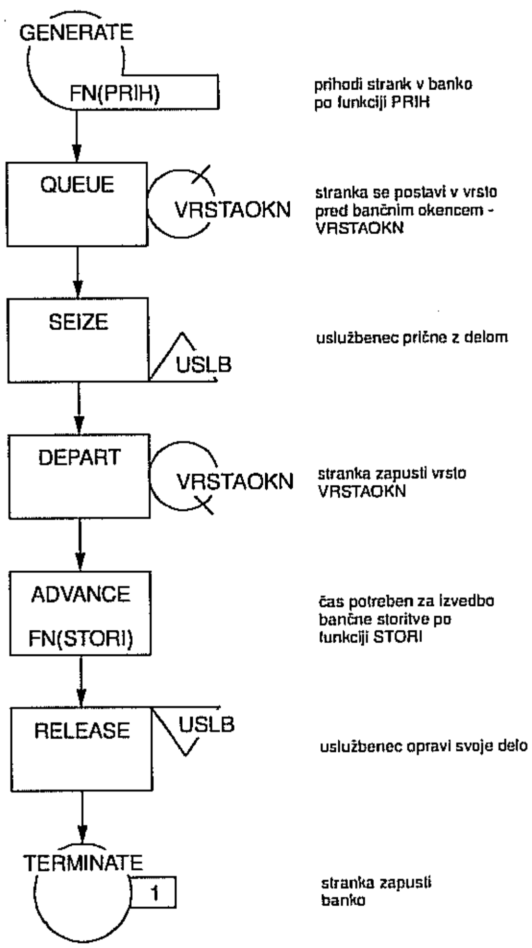
Pri simulaciji strežnih sistemov nas zanima število ljudi v vrsti (dolžina vrste), čas čakanja na strežbo, čas strežbe, zasedenost strežnega mesta itd. Te lastnosti so odvisne od hitrosti prihoda strank v banko, časa opravljanja bančne storitve (strežbe) in vrste časovnih distribucij.

Naloga

Cilj naloge je pokazati podobnost rezultatov in enostavnost reševanja problemov v jeziku GPSS. Iz tega razloga bomo uporabili iste podatke, tudi dolžino simulacijskega teka.

Blokovni diagram

Slika 29 prikazuje blokovni diagram strežbe v banki.



Slika 29: Blokovni diagram strežbe v banki

Simulacijski model

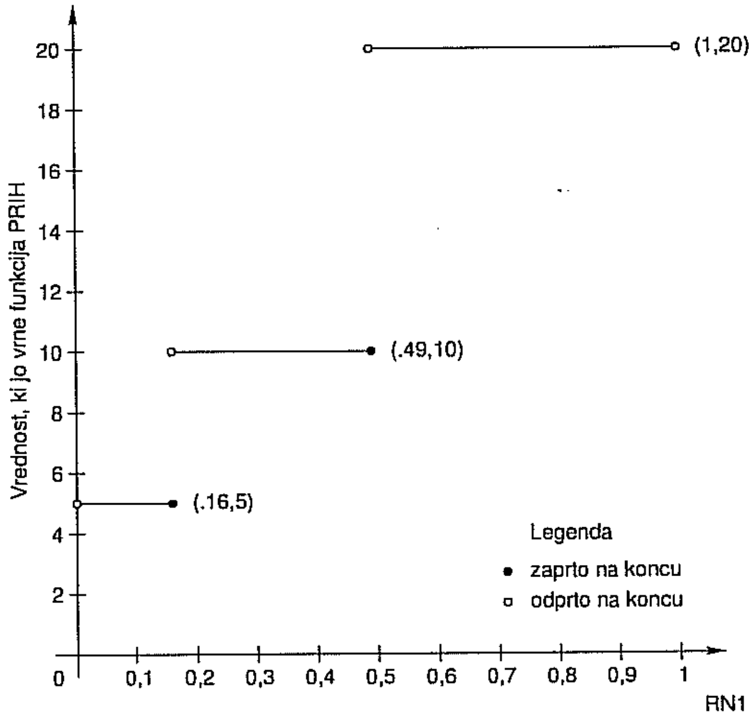
Simulacijski model v GPSS/H je prikazan na Slika 30. Opazna je podobnost med blokovnim diagramom in simulacijski ukazi. Blok GENERATE generira prihod strank v banko. Le te smo definirali s funkcijo PRIH. To je ukaz iz SNA. FUNCTION je rezervirana beseda, ki napove uporabo empirične funkcije, RN(1) predstavlja prvi generator naključnih števil enakomerne distribucije, D4 pa pomeni da čas prihoda strank vzamemo kot diskretni proces. Zaradi tega lahko primerjamo rezultate ročne simulacije, kjer smo za generator naključnih števila vzeli pravo kocko. Kumulativne rezultate smo zaokrožili. Enako smo naredili za čas strežbe, ki je definirana s funkcijo STORI, uporabljen pa je drugi naključni generator - RN(2). Ostali del programa je podoben kot pri primeru luke.

```
        SIMULATE
*
*       Model banke s strežnim mestom in vrsto
*
*       Definiramo funkcijo PRIH - čas prihoda
*
PRIH    FUNCTION RN(1),D4
0,0/.16,5/.49,10/1,20
*
*       Definiramo funkcijo STORI - čas izvajanja bančne storitve
*
STORI   FUNCTION RN(2),D4
0,0/.16,10/.33,5/1,15
*
*
*       GPSS/H blokovni del
*
        GENERATE   FN(PRIH)          Prihodi strank v banko
        QUEUE      VRSTAOKN          Stranka se postavi v vrsto pred bančnim okencem
        SEIZE      USLB              Uslužbenec prične z delom
        DEPART     VRSTAOKN          Stranka zapusti vrsto
        ADVANCE    FN(STORI)         Izvajanje bančne storitve
        RELEASE    USLB              Uslužbenec opravi storitev
        TERMINATE  1                 Stranka zapusti banko
*
*       GPSS/H kontrolni stavki
*
        START      10                TG1 (terminations to go) postavimo na 10
        CLEAR                        Reset statistike, transakcije iz modela
        START      10                TG1 (terminations to go) postavimo na 10
        CLEAR                        Reset statistike, transakcije iz modela
        START      10                TG1 (terminations to go) postavimo na 10
        CLEAR                        Reset statistike, transakcije iz modela
        START      10                TG1 (terminations to go) postavimo na 10
        CLEAR                        Reset statistike, transakcije iz modela
        START      10                TG1 (terminations to go) postavimo na 10
        CLEAR                        Reset statistike, transakcije iz modela
        START      10                TG1 (terminations to go) postavimo na 10
        CLEAR                        Reset statistike, transakcije iz modela
        START      10                TG1 (terminations to go) postavimo na 10
        CLEAR                        Reset statistike, transakcije iz modela
        START      10                TG1 (terminations to go) postavimo na 10
        CLEAR                        Reset statistike, transakcije iz modela
        START      10                TG1 (terminations to go) postavimo na 10
        CLEAR                        Reset statistike, transakcije iz modela
        START      10                TG1 (terminations to go) postavimo na 10
        END
```

Slika 30: Simulacijski model za primer strežbe v banki

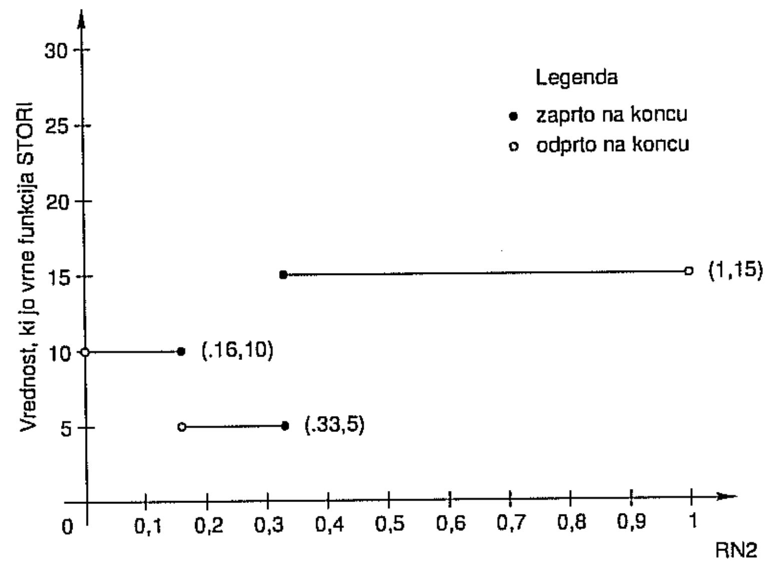
Način generiranja prihoda in strežbe strank v funkciji naključnih generatorjev prikazujeta Slika 7 in Slika 8. Sliki prikazujeta funkcijo števila prihodov (PRIH) oziroma časa strežbe (STORI), ki jo oblikujemo s pomočjo naključnega generatorja števil. Na abscisi (os x) je podana vrednost števila, ki ga generira naključni generator, na ordinati (os y)pa je podano število prihodov oziroma časa strežbe, ki temu številu ustreza. Če generator naključnih števil vrne vrednost na intervalu, npr. 0.16, potem se vzame vrednost zgornjega-zaprtega intervala. Npr. za vrednost 0.49, funkcija vrne vrednost 10.

Slika 31 prikazuje grafično interpretacijo funkcije prihoda strank v banko.



Slika 31: Grafična interpretacija funkcije prihoda v banko

Slika 32 prikazuje grafično interpretacijo funkcije časov izvajanja bančne storitve oz. časov strežbe v banki.

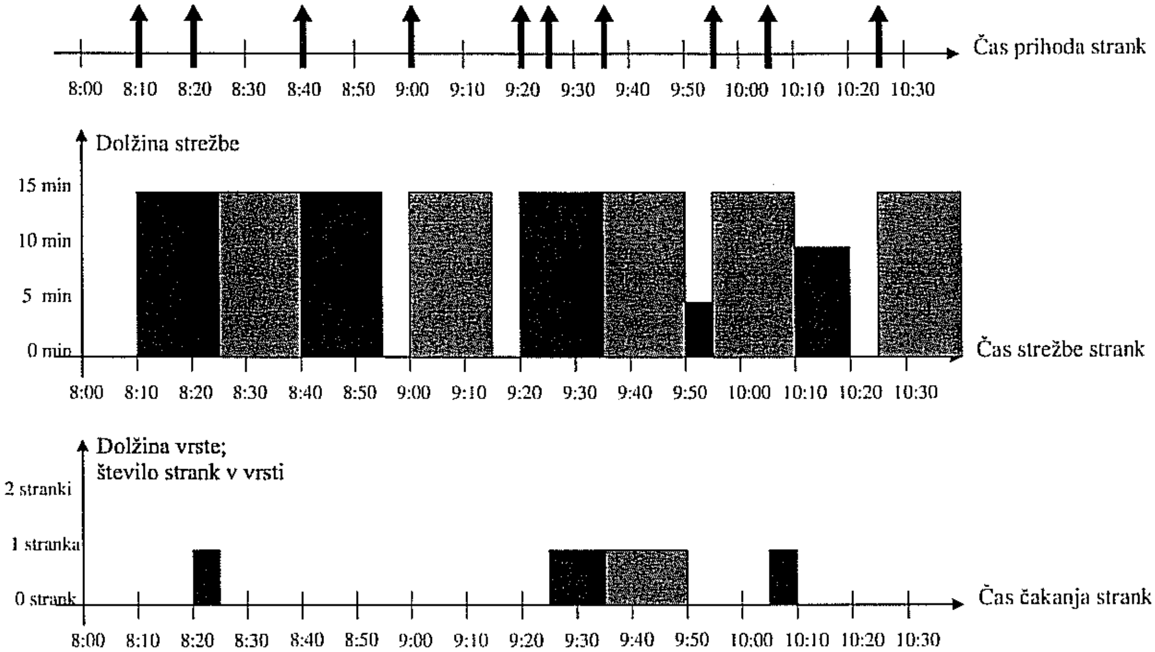


Slika 32: Grafična interpretacija funkcije časov strežbe v banki

Na koncu smo pod kontrolne stavke dali ukaze START 10, CLEAR, ki se ponavljajo 10 krat. Ukaz START 10 pomeni simuliraj do 10 stranki, Ukaz CLEAR pa pomeni da po izpisu rezultatov izbriše vso statistiko in transakcije iz modela, razen zadnje vrednosti naključnega generatorja. To pomeni, da se naslednji simulacijski tek ne začne s semenom generatorja marveč z zadnjo uporabljeno vrednostjo generatorja naključnih števil. Na ta način dobimo nov, neodvisni tek. S tem se želimo prepričati ali so rezultati simulacije stacionarni, oz. če so si med sabo podobni.

Rezultati

Slika 33 grafično prikazuje simulacijske rezultate časov, ki jih zbere statistika v GPSSu.



Slika 33: Grafični prikaz rezultatov časa prihoda strank in strežbe, število strank v vrsti in dolžino v banki.

Rezultate posamičnih tekov smo zbrali v tabelah 10 in 11. Za pravo vrednost izračunamo povprečje simulacijskih tekov (npr. Povprečni čas strežbe stranke = 12,30) in njihovo standardno deviacijo. Tabela 10 prikazuje simulacijske rezultate za delovno mesto bančnega uslužbenca - USLB.

Tabela 10: Simulacijski rezultati za delovno mesto USLB

FACILITY USLB

Zaporedna številka simulacijskega teka	Celotni povprečni delež časa strežbe	Povprečni čas strežbe stranke
1	0,88	14,00
2	0,71	13,50
3	0,60	10,50
4	0,72	11,50
5	0,76	12,50
6	0,87	13,50
7	0,89	12,00
8	0,71	11,00
9	0,79	13,00
10	0,61	11,50
SUM	7,53	123,00
AVG	0,75	12,30

Zgornja tabela prikazuje povprečne čase posameznih tekov simulacije zaradi ukaza START in CLEAR. Jasno je videti, da deleži časov strežbe in povprečni časi strežbe niso enaki. Prihaja do precej velikih razlik, kar je posledica uporabe drugih naključnih števil in verjetnostnih funkcij v modelu. Če bi za referenčne vzeli rezultate samo enega teka, bi dobili bistveno drugačne rezultate kot so le ti po opravljenih več tekih simulacije. Očitno deset transakcij, s katerimi poženemo model (START 10) ni dovolj, da bi model prišel v stacionarno fazo, kjer bi bili rezultati blizu povprečnih. Pri eksperimentiranju na modelu je tako potrebno paziti, da izvajamo eksperiment z zadostnim številom transakcij, da model pride v stacionarno fazo, ali pa izvajamo več ponovitev tekov in analiziramo povprečne rezultate.

Tabela 11: prikazuje simulacijske rezultate za vrsto pred bančnim okencem - VRSTAOKN.

QUEUE VRSTAOKN

Zaporedna številka sim. teka	Maksimalno število strank v vrsti	Povprečno število strank v vrsti	Skupno število prispelih strank	Število strank, ki jim ni potrebno čakati	Povprečni čas čakanja vseh strank	Povprečni čas čakanja strank, ki dejansko čakajo	Trenutno število strank v vrsti
1	2,00	0,59	12,00	3,00	7,92	10,56	2,00
2	1,00	0,05	11,00	9,00	0,91	5,00	1,00
3	1,00	0,11	10,00	7,00	2,00	6,67	0,00
4	1,00	0,13	11,00	8,00	1,82	6,67	1,00
5	1,00	0,12	10,00	7,00	2,00	6,67	0,00
6	1,00	0,29	11,00	6,00	4,09	9,00	1,00
7	1,00	0,22	11,00	5,00	2,73	5,00	1,00
8	1,00	0,13	10,00	8,00	2,00	10,00	0,00
9	1,00	0,12	10,00	6,00	2,00	5,00	0,00
10	1,00	0,00	10,00	10,00	0,00	0,00	0,00
SUM	11,00	1,77	106,00	69,00	25,46	64,56	6,00
AVG	1,10	0,18	10,60	6,90	2,55	6,46	0,60

V primeru, da eksperimentiramo kaj se bo zgodilo pri krajšem času izvajanja bančne storitve in uvedemo krajši čas po funkciji:

```
STORI   FUNCTION RN(2),D4
0,0/.16,5/.33,2/1,7
```

nam
Tabela 12 in Tabela 13 prikazujeta simulacijske rezultate desetih tekov in povprečja.

Tabela 12: Del simulacijskih rezultatov za delovno mesto USLB – krajši čas izvajanja storitve

FACILITY USLB

Zaporedna številka simulacijskega teka	Celotni povprečni delež časa strežbe	Povprečni čas strežbe stranke
1	0,45	6,50
2	0,37	6,40
3	0,29	4,80
4	0,37	5,40
5	0,39	5,90
6	0,40	6,30
7	0,43	5,60
8	0,35	5,10
9	0,39	6,10

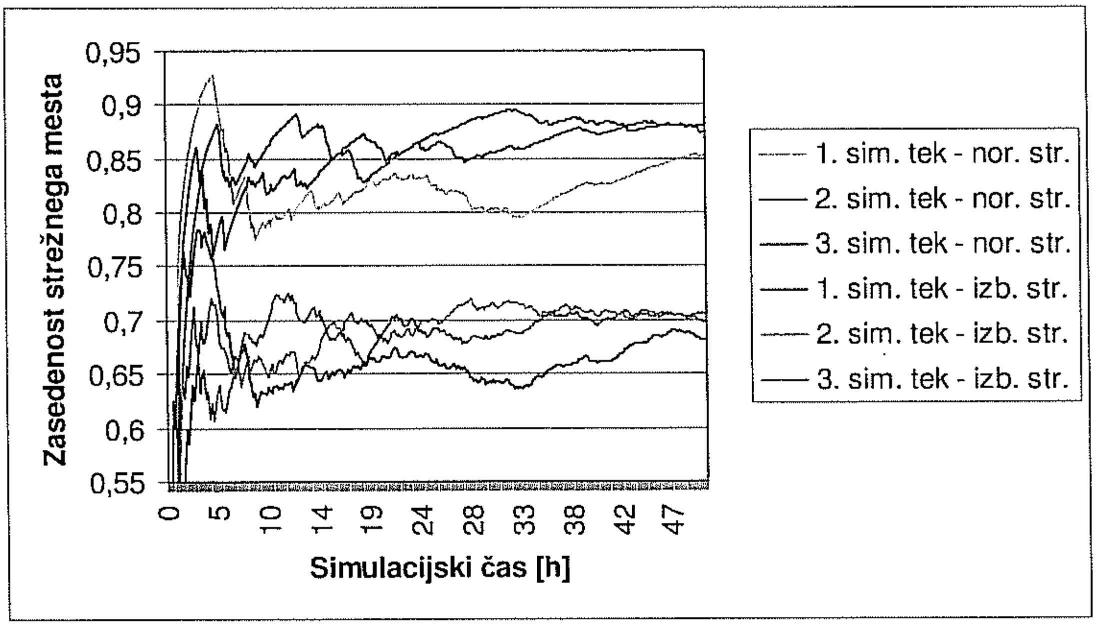
10	0,33	5,40
SUM	3,77	57,50
AVG	0,38	5,75

Tabela 13: Del simulacijskih rezultatov za vrsto VRSTAOKN – krajši čas izvajanja storitve

QUEUE VRSTAOKN

Zaporedna številka sim. teka	Maksimalno število strank v vrsti	Povprečno število strank v vrsti	Število strank, ki jim ni potrebno čakati	Povprečni čas čakanja vseh strank	Povprečni čas čakanja strank, ki dejansko čakajo
1	1,00	0,03	8,00	0,40	2,00
2	1,00	0,00	10,00	0,00	0,00
3	1,00	0,01	9,00	0,20	2,00
4	1,00	0,00	10,00	0,00	0,00
5	1,00	0,00	10,00	0,00	0,00
6	1,00	0,03	8,00	0,40	2,00
7	1,00	0,02	9,00	0,20	2,00
8	1,00	0,01	9,00	0,20	2,00
9	1,00	0,04	8,00	0,60	3,00
10	1,00	0,01	9,00	0,20	2,00
SUM	10,00	0,15	90,00	2,20	15,00
AVG	1,00	0,01	9,00	0,22	1,50

Slika 34 prikazuje graf za normalno ter izboljšano strežbo (normalna – 10, 5, 15 / izboljšana – 8, 4, 12) za čas simulacije petdeset ur. Na abscisi je podan simulacijski čas v urah, medtem ko ordinata prikazuje zasedenost strežnega mesta pri eksperimentiranju z osnovno in izboljšano funkcijo strežbe.



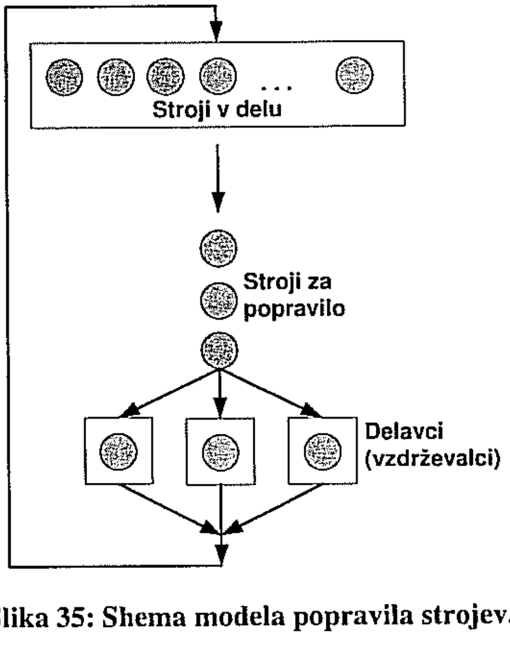
Slika 34: Zasedenost strežnega mesta

Rezultati simulacije oz. stacionarnost je odvisna od razmerja hitrosti prihoda strank v sistem in hitrosti strežbe. V prvem primeru v tabelah 1 in 2 so za $\rho=12.5/14.16 = 0.85$. Rezultati simulacije za razmerje obremenitve $\rho= 0.41$ pa so prikazani v tabelah 3 in 4. Seveda je simulacija do postreženih 10 strank tudi za tako enostaven sistem premalo. Sistem se še ni postavil v ravnotežno stanje, rezultati strežbe, ki smo jih dobili kot povprečje 10 neodvisnih tekov in so precej razpršeni zaradi efekta prehodnega pojava. Bolj natančne rezultate bi dobili za simulacijo več strank. Efekt dolžine simulacije na stacionarnost sistema kaže Slika 34, kjer smo simulirali delovni čas 50h. Slika prikazuje povprečje zasedenosti uslužbenca in povprečja parametrov vrste ki smo jih izračunavali vsakih 10 minut. Za vsako obremenitev smo simulirali tri neodvisne teke. Vidimo, da povprečja konvergirajo proti neki povprečni vrednosti šele po 50h simulacije. Sistem vzamemo za stacionarni, ko je čas procesa dovolj dolg. Sicer pri stacionarnosti procesa ne razumemo le, da se ustali povprečje, marveč tudi, da je varianca procesa neodvisna od časa. Veliko strežnih sistemov nikoli ne doseže stacionarnega stanje. Na primer, če banka dela 8 h, mora tudi simulacija trajati 8 simulacijskih ur. V tem primeru imamo opravka s terminalnim strežnim sistemom. Pravo vrednost simulacije dobimo z večkratnim ponavljanjem simulacije (z ukazom START, CLEAR) kot v pravkar obravnavanem primeru.

7.3 Model popravila strojev

Definicija modela

V tkalski delavnici so instalirani stroji, katerim po določenem času obratovanja se potrga nitka in stroj ni v obratovanju. Zaradi tega pride do izpada proizvodnje. Najbližji delavec - vzdrževalec gre k stroju in napako odpravi. Gre za zaprti sistem strežbe, kjer stroji predstavljajo transakcije. Število strojev je konstantno; to je število strojev v delu + število strojev, ki čakajo na popravilo. V delavnici je 15 tkalskih strojev, povprečni čas obratovanja stroja do okvare je 300 sek, porazdeljenih po eksponentni distribuciji. Čas popravila stroja je 60 sek, z odklonom 60 sek (60±60 sek).



Slika 35: Shema modela popravila strojev.

Problem

Potrebno je določiti število vzdrževalcev tako da je izguba zaradi zastoja čim manjša pri primernih stroških osebnih dohodkov vzdrževalcev ob upoštevanju ergonomskih predpisov obremenitve delavcev vzdrževalcev. Delo na vzdrževanju je naporno in je ta omejitev pomembna.

Naloga

Ugotoviti najprimernejše število delavcev za vzdrževanje tkalskih strojev, da so stroški in izguba čim manjša ob upoštevanju ergonomskih predpisov.

Kriterij

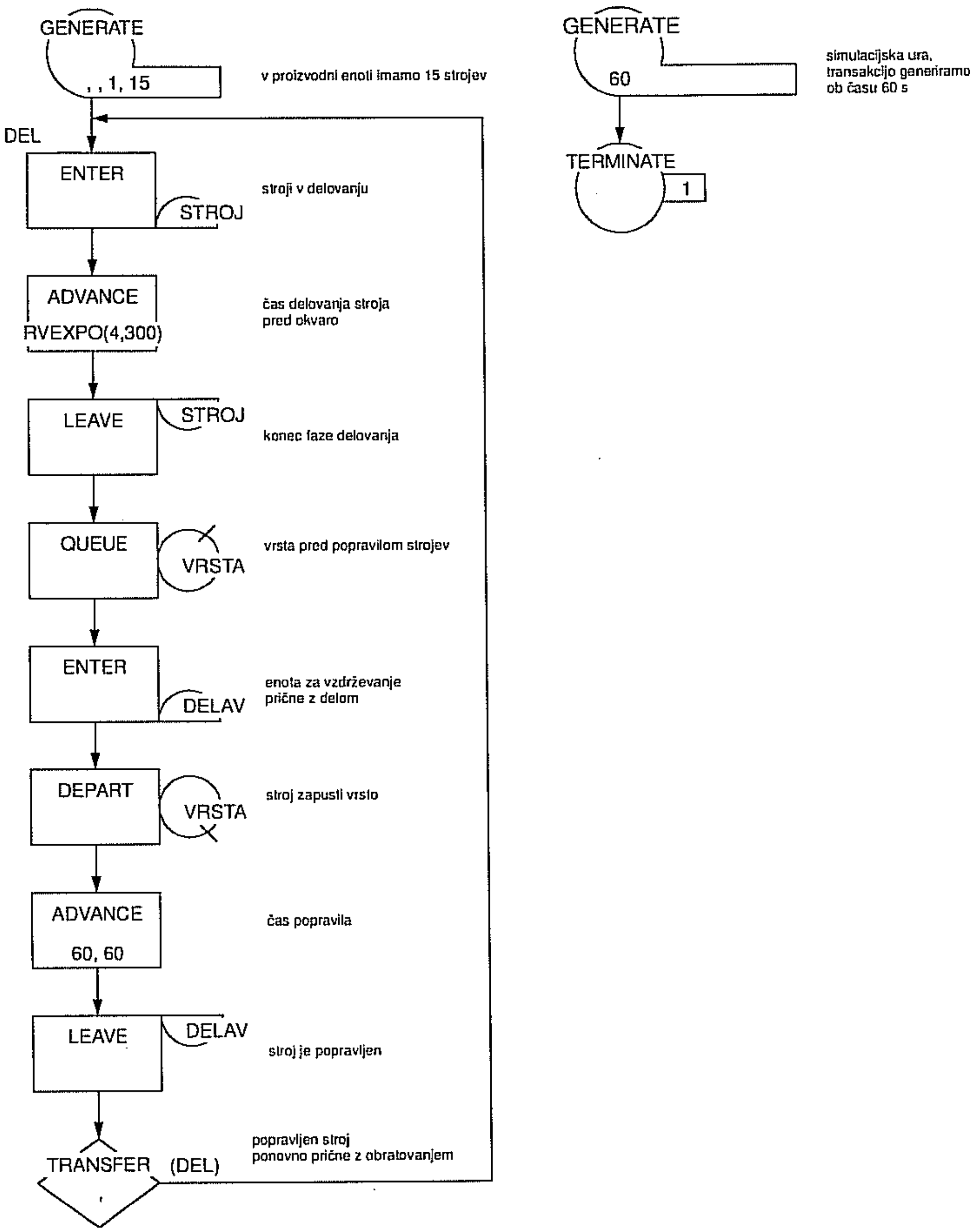
$$\min I(\eta, N) = (1 - \eta) * D * R + C\_del * N,$$

$$Q = (1 - \eta) * R * D$$

$$P = N * C\_del$$

kjer je:
I = celotna izguba
Q = izguba zaradi neizkoriščenosti strojev
η = izkoristek stroja
D = dnevni dobiček na stroju (150000)
R = število strojev (15)
C_del = dnevni stroški na delavca (8000)
P = plača delavcev
N = število delavcev – vzdrževalcev strojev.

Blokovni diagram



Slika 36: Blokovni diagram za model popravila strojev

Simulacijski model

```
        SIMULATE
*
*       Simulacija popravila strojev
*
*       Definicija spremenljivke za eksperimentiranje
        INTEGER &KAP
        PUTPIC
        VNESITE ŠTEVILO DELAVCEV:
        GETLIST &KAP
*
*       Deklaracija kapacitet
        STORAGE  S(STROJ),15/S(DELAV),&KAP
*
*       GPSS/H blokovni del
*
        GENERATE        ,,1,15           V času 1 generiramo 15 transakcij
DEL     ENTER           STROJ            Vstop v stroj
        ADVANCE         RVEXPO(4,300)    Delo na stroju
        LEAVE           STROJ            Delo na stroju je končano
        QUEUE           VRSTA            Vrsta pred dodelavo
        ENTER           DELAV            Vstop v delavnico
        DEPART          VRSTA            Zapustimo vrsto
        ADVANCE         60,60            Ročno delo
        LEAVE           DELAV            Zapustimo delavnico
        TRANSFER ,DEL                    Ponovimo postopek
*
*       GPSS/H kontrolni stavki
*       Simulacija na čas
*
        GENERATE 60                      Generiramo transakcijo vsakih 60 časovnih enot
        TERMINATE       1                Transakcija zapusti model
        START           2400             Čas simulacije: 40h
        END
```

Slika 37: Program v jeziku GPSS za model popravila strojev

Za potrebe eksperimentiranja smo v simulacijski model vnesli spremenljivko, katere vrednost vnašamo pri vsakem zagonu simulacijskega programa. Predstavlja nam število delavcev-vzdrževalcev, ki popravljajo stroje.

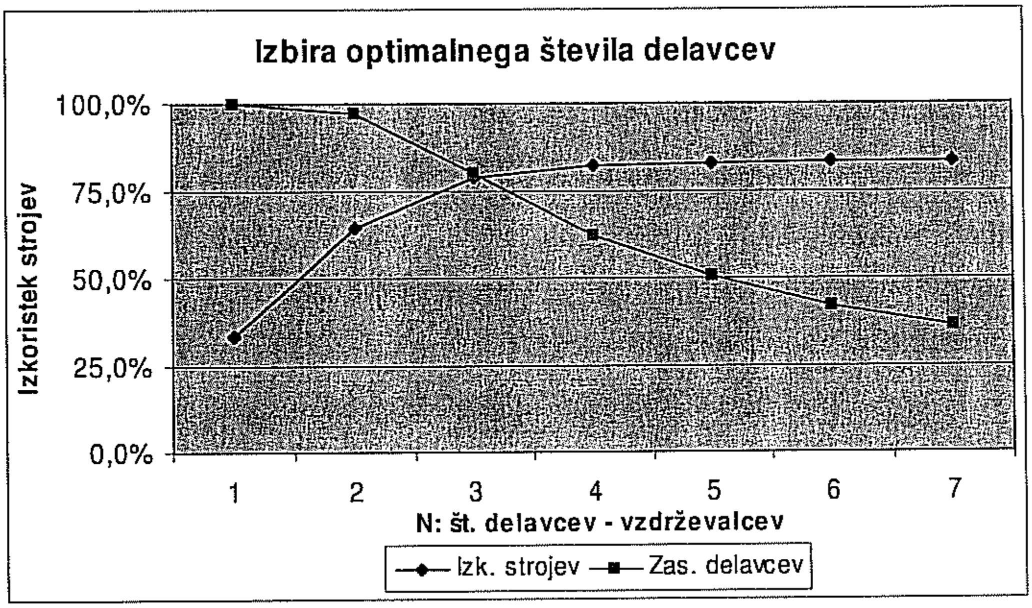
Najprej moramo definirati spremenljivko, ki je v našem primeru celoštevilčna vrednost z imenom KAP (INTEGER &KAP). Oznaka & je dodana kot ločilo med napovedjo tipa in imenom spremenljivke. Naslednji je ukaz PUTPIC, ki ob zagonu simulacijskega modela na zaslon izpiše tekst, ki je zapisan v naslednji vrstici programa (VNESITE STEVILO DELAVCEV:). Po vnosu spremenljivke je le to potrebno prebrati v model. To storimo z ukazom GETLIST in imenom spremenljivke (&KAP). Vrednost spremenljivke, ki jo uporabnik vpiše ob zagonu modela, se pri deklaracijah kapacitet pripiše kapaciteti delavcev z oznako DELAV, kar storimo z ukazom STORAGE 15,/S(DELAV),&KAP.

Rezultat

Rezultati simulacije so zbrani v Tabela 14, grafični prikaz pa je podan na Slika 38.

Tabela 14: Odvisnost izkoriščenosti strojev od števila delavcev

Število delavcev	Izkoriščenost strojev	Zasedenost delavcev
1	33,4%	100,0%
2	64,4%	97,1%
3	78,6%	79,8%
4	82,1%	62,2%
5	82,9%	50,4%
6	83,1%	42,2%
7	83,1%	36,2%

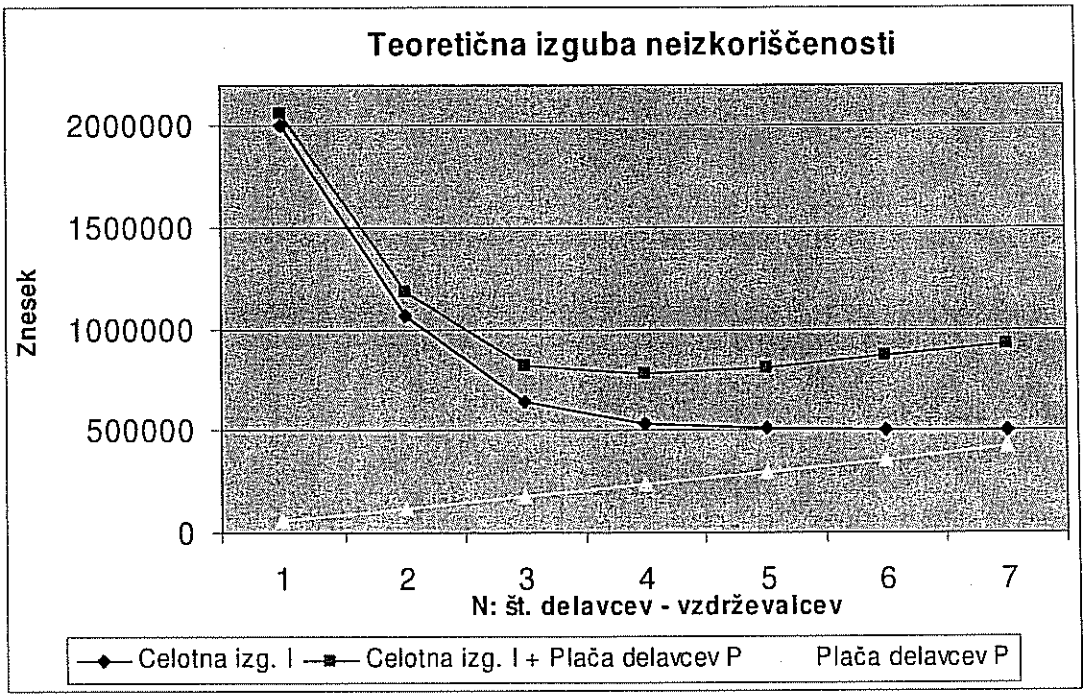


Slika 38: Izkoriščenost strojev in zasedenost delavcev v funkciji števila vzdrževalcev

Na podlagi podatkov v Tabela 14in kriterijske funkcije lahko izračunamo vrednosti celotne izgube ter njenih komponent; stroškov in plače delavcev, ki jih prikazuje Tabela 15. Graf Tabela 15 je prikazan na Slika 39. Pri izbranih vrednosti plač delavcev ter vrednosti izpada proizvodnje strojev je primerno število delavcev štiri. Pri tem je obremenitev delavcev 62%, kar je še vedno pod dovoljeno normo.

Tabela 15: Celotna izguba v odvisnosti od plač delavcev (v SIT)

Št. delavcev	Celotna izg. I	Izguba I + Izg. neizkoriščenost Q	Plača delavcev P
1	2028000	2058000	30000
2	1128000	1188000	60000
3	732000	822000	90000
4	657000	777000	120000
5	663000	813000	150000
6	687000	867000	180000
7	717000	927000	210000



Slika 39: Odvisnost izgub podjetja v funkciji števila delavcev.

V Tabela 15 smo podali rezultate simulacije samo enega teka. Postavi se vprašanje zanesljivosti rezultatov, glede na to, da imamo opravka z stohastičnim procesov. V prejšnji nalogi smo pokazali variabilnost rezultatov pri različnih simulacijskih tekih. Zaradi tega bomo rezultate simulacije za posamezno število delavcev ponovili večkrat pri različnih začetnih vrednosti naključnega generatorja (uporabili bomo ukaz START in CLEAR). Rezultati povprečne izkoriščenosti strojev in standardne devijacije simulacije deset tekov so tako podani v Tabela 16.

Tabela 16: Povprečna izkoriščenost strojev s standardno deviacijo za deset simulacijskih tekov

Število mehanikov	povprečje 10 tekov	standardna deviacija
2	0,654	0,029
3	0,789	0,010
4	0,820	0,004
5	0,831	0,003

Kot je razvidno, zaposlitev petega vzdrževalca dvigne izkoriščenost strojev za dodatni 1%. Ob ustrezni ceni strojne ure in plači delavcev zna biti odločitev za pet vzdrževalcev pravilna, seveda pod pogojem da 1% izboljšave ni zgolj slučaj. Zaradi tega skušamo potrditi nično hipotezo H_0 in ugotoviti ali je razlika izboljšave za 1% pri zaposlitvi petih delavcev pomembna. Hipotezo testiramo s t-testom po formuli:

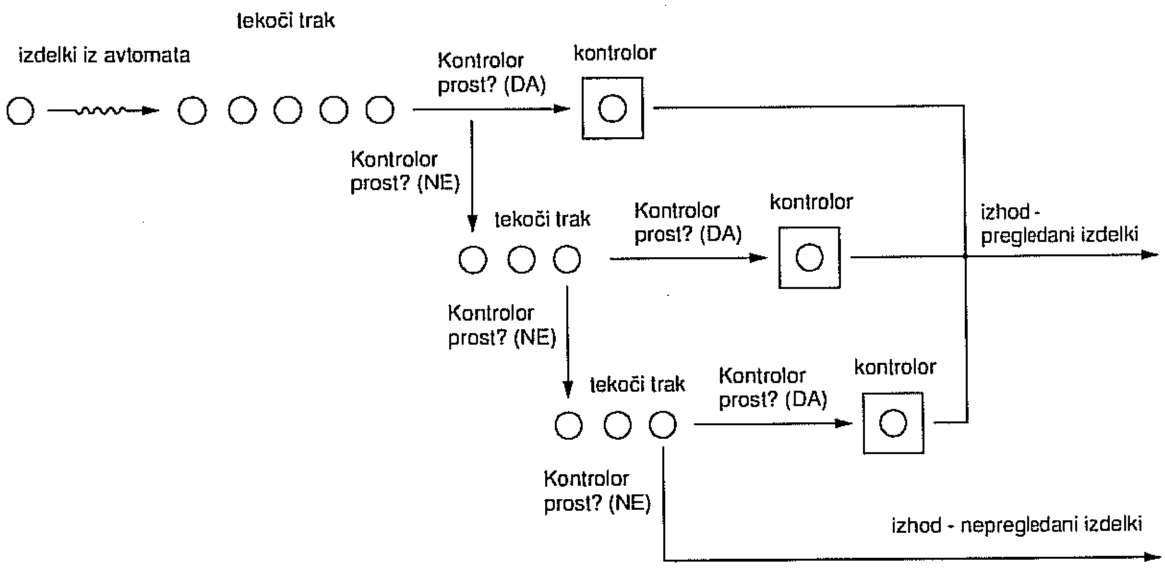
$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\dfrac{n_1\sigma_1^2 + n_2\sigma_2^2}{n_1 + n_2}}} \sqrt{\frac{n_1 n_2}{n_1 + n_2}}$$

kjer sta $\overline{x}_1, \overline{x}_2$ povprečja izkoriščenosti strojev za štiri in pet delavcev in σ_1, σ_2 ustrezne standardne devijacije. Z vstavitvijo posameznih vrednosti v formulo dobimo vrednost $t(4{,}5) = 12{,}6$. V tabeli za t statistiko pri izbranem tveganju $p = 0{,}01$ in stopnji prostosti $\nu = 18$, (n1+n2-2=18), preberemo kritično vrednost $t_c = 3{,}33$. Ker je izračunani $t(4{,}5) >> t_c$ smemo trdit da je zgolj 1% izkoriščenosti strojev je pomembno tehnološko izboljšanje, ki se ekonomsko ne izplača. Lahko bi bilo tudi nasprotno: da je tehnološka izboljšava posledica naključja, čeprav je ekonomsko upravičena.

7.4 Delavnica 1

Definicija modela

Imamo primer kontrole izdelkov v proizvodnji, kjer trije kontrolorji pregledujejo dokončane izdelke pred pakiranjem. Izdelki prihajajo po tekočem traku, najprej mimo kontrolorja 1. Da izdelek prispe do prvega kontrolorja traja dve minuti. Če je prost, ko izdelek prispe do njega, ga vzame v pregled. V nasprotnem primeru, izdelek potrebuje nadaljnji dve minuti, da prispe do kontrolorja 2, ki vzame izdelek v pregled v primeru, da je prost. Izdelek gre tudi mimo drugega kontrolorja, če le ta ni prost. Po nadaljnjih dveh minutah doseže tretjega kontrolorja. V primeru, da je zaseden tudi ta kontrolor, izdelek ni pregledan in gre na pakiranje. Izdelki na začetek tekočega traku prihajajo povprečno vsakih pet minut z enakomernim odmikom treh minut. Inšpektorji za pregled posameznega izdelka potrebujejo v povprečju dvanajst minut z enakomernim odmikom devet minut.



Slika 40: Shema modela predstavljene delavnice.

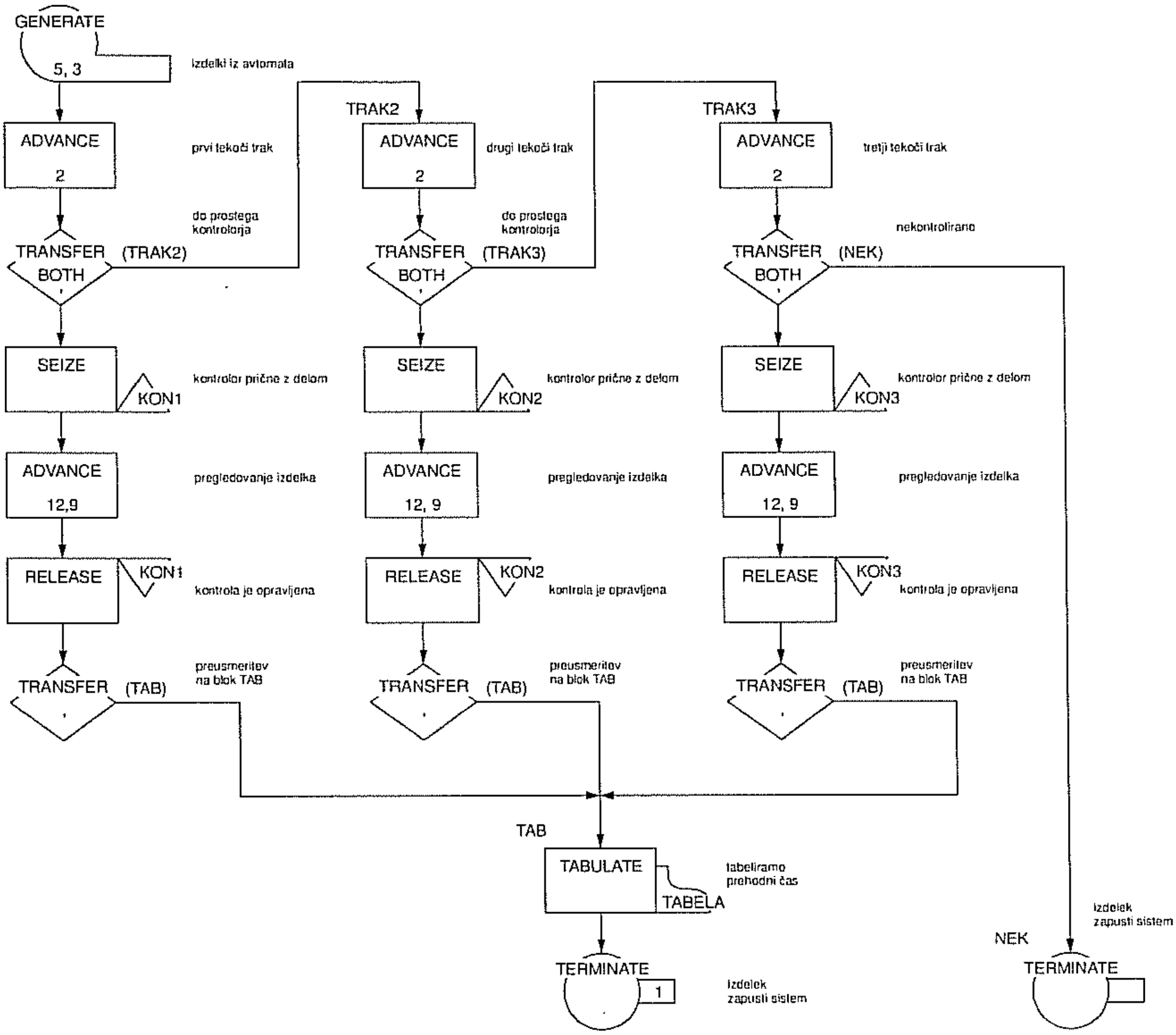
Problem

Ker želimo zagotoviti sto odstotno kontrolo izdelkov, je potrebno ugotoviti, ali kontrola izvedena na predstavljeni način zmore pregledati vse izdelke. Poleg tega bi želeli zagotoviti enake delovne pogoje vsem kontrolorjem, zato bi bil zanimiv tudi podatek o zasedenosti kontrolorjev.

Naloga

Modeliraj predstavljeni sistem za tisoč izdelkov. Ugotoviti je potrebno, ali so pregledani vsi izdelki, v primeru pa, da kontrolorji ne uspejo pregledati vseh izdelkov je potrebno ugotoviti delež nepregledanih. Poleg tega je potrebno ugotoviti tudi obremenjenost posameznega kontrolorja in število izdelkov, ki jih vsak izmed njih pregleda.

Blokovni diagram



Slika 41: Blokovni diagram za model delavnice

Simulacijski model

Blokovni diagram zgoraj opisanega sistema prikazuje Slika 41.

```
         SIMULATE
*
*        Simulacija kontrole izdelkov
*        Osnovna časovna enota: 1 minuta
*
*
*        Definiramo tabelo
*
TABELA   TABLE M1,5,5,10
*
```

```
*       GPSS/H blokovni del
*
*
        GENERATE     5,3              Izdelki prihajajo iz avtomata v času 5 +- 3 min
*
*       Prvi tekoči trak in kontrolor
*
        ADVANCE      2                Tekoči trak 1
        TRANSFER     BOTH,,TRAK2      Do prvega prostega kontrolorja
        SEIZE        KON1             Prvi kontrolor je zaseden
        ADVANCE      12,9             Pregledovanje izdelka
        RELEASE      KON1             Prvi kontrolor je prost
        TRANSFER     ,TAB             Na blok TAB
*
*       Drugi tekoči trak in kontrolor
*
TRAK2   ADVANCE      2                Tekoči trak 2
        TRANSFER     BOTH,,TRAK3      Do prvega prostega kontrolorja
        SEIZE        KON2             Drugi kontrolor je zaseden
        ADVANCE      12,9             Pregledovanje izdelka
        RELEASE      KON2             Drugi kontrolor je prost
        TRANSFER     ,TAB             Na blok TAB
*
*       Tretji tekoči trak in kontrolor
*
TRAK3   ADVANCE      2                Tekoči trak 3
        TRANSFER     BOTH,,NEK        Ali je tretji kontrolor prost
        SEIZE        KON3             Tretji kontrolor je zaseden
        ADVANCE      12,9             Pregledovanje izdelka
        RELEASE      KON3             Tretji kontrolor je prost
        TRANSFER     ,TAB             Na blok TAB
*
*       Nekontrolirani izdelki
*
NEK     TERMINATE
*
TAB     TABULATE     TABELA           V tabelo zapišemo prehodni čas transakcije
        TERMINATE    1
*
*       GPSS/H kontrolni stavki
*
        START        10,NP            TG1 (terminations to go) postavimo na 10 - brez izpisa
        RESET                         Resetiramo vso statistiko
        START        1000             TG1 (terminations to go) postavimo na 1000
        END
```

Slika 42: Program v jeziku GPSS za model delavnice

Premikanje izdelkov po tekočem traku je modelirano z blokom ADVANCE, kjer je čas potovanja dve minuti. Ko transakcija zapusti ADVANCE blok, preveri v bloku TRANSFER, če je kontrolor prost z selekcijskim faktorjem BOTH. Izhod iz vsakega od bloka TRANSFER vodi v blok SEIZE, ki predstavlja zasedbo posameznega kontrolorja. Če je kontrolor prost v času ko transakcija vstopi v blok TRANSFER, zapusti blok skozi izhod 1 (torej nadaljuje pot skozi strežbo), v primeru pa, da je zaseden gre transakcija skozi izhod 2 (skoči na blok ADVANCE 2 in potem na drugi blok TRANSFER).

Ko so izdelki pregledani, transakcija nadaljuje v blok TABULATE, kjer se zapiše prehodni čas transakcije. Prvi ukaz START požene model za deset strank, kjer se rezultati ne izpišejo v statistiko (NP). Zatem uporabimo kartico RESET, kjer resetiramo vso statistiko, transakcije, ki se nahajajo v modelu pa v njem ostanejo. Zatem simuliramo model za tisoč strank.

Rezultati

RELATIVE CLOCK: 5931.6567 ABSOLUTE CLOCK: 6012.8959

BLOCK	CURRENT	TOTAL	BLOCK	CURRENT	TOTAL	BLOCK	CURRENT	TOTAL
1		1178	11	1	351	TAB		1000
2	1	1178	12		350	22		1000
3		1177	13		350			
4		397	TRAK3		429			
5		397	15		429			
6		398	16		253			
7		398	17	1	253			
TRAK2		780	18		252			
9		780	19		252			
10		351	NEK		176			

FACILITY	--AVG-UTIL-DURING-- TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/XACT	CURRENT STATUS	PERCENT AVAIL	SEIZING XACT	PREEMPTING XACT
KON1	0.815			398	12.143	AVAIL			
KON2	0.711			351	12.009	AVAIL		1194	
KON3	0.513			253	12.029	AVAIL		1193	

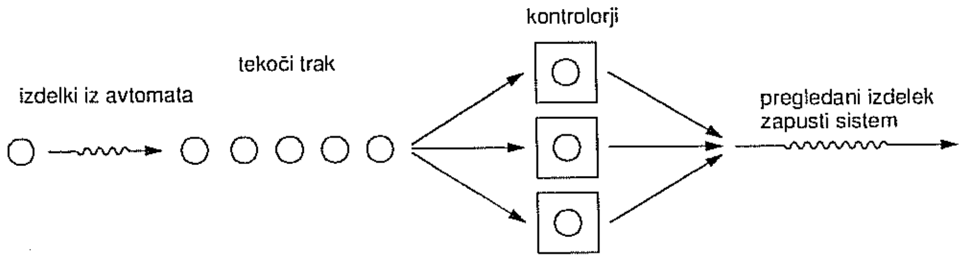
Slika 43: Delni rezultati simulacije za model delavnice

Delni rezultati, ki jih prikazuje Slika 43 kažejo, da je zasedenost posameznih kontrolorjev neenakomerna (KON1 = 81,5% časa, KON2 = 71,1% časa, KON3 = 51,3% časa), kar je posledica zaporedne postavitve kontrolorjev. Poleg tega kontrolorji ne uspejo pregledati vseh proizvedenih izdelkov, saj je razvidno, da skozi del modela NEK potuje 176 nepregledanih izdelkov na vsakih 1000 pregledanih. Število izdelkov, ki jih vsak izmed kontrolorjev pregleda pa je v sorazmerju z njegovo zasedenostjo.

7.5 Delavnica 2

Definicija modela

Predstavljen je model preproste, avtomatizirane proizvodnje, kjer izdelki po tekočem traku prihajajo pred končno kontrolo. Trije kontrolorji opravljajo pregled izdelkov in jih označijo glede na njihove kvalitativne lastnosti. Po pregledu jih spustijo naprej po tekočem traku v nadaljnjo obdelavo. V programu tabeliramo podatke o času pregleda izdelkov, ki jih posamezni kontrolor porabi za kontrolo izdelka. Časi prihodov in pregledovanja so isti kot pri zgornjem primeru delavnice 1.



Slika 44: Delni rezultati simulacije za model delavnice

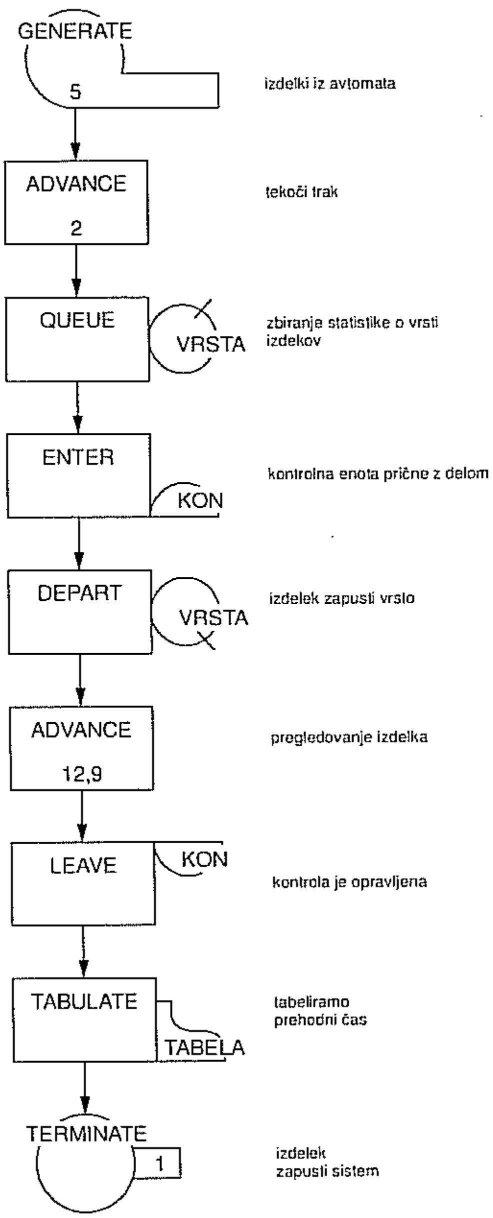
Problem

Potrebno je določiti zasedenost kontrolorjev pri pregledu izdelkov, saj je kakovost pregledovanja odvisna od zasedenosti. Namreč, če so kontrolorji zasedeni prevelik odstotek časa, je kakovost pregleda zaradi hitenja in utrujenosti vprašljiva. Pri premajhni zasedenosti kontrolorjev pa pride do obratnega učinka, saj kakovost pregleda upada, ker se kontrolorji ne poglobijo dovolj v pregled, hkrati pa njihova koncentracija pri pregledovanju ni dovolj visoka in delo opravijo površno. Ugotoviti je potrebno tudi dolžino vrste izdelkov za pregled in ali je sistem propusten, torej ali kontrolorji uspejo pregledati vse izdelke.

Naloga

Modeliraj predstavljeni sistem za tisoč izdelkov. Ugotoviti je potrebno, ali je sistem pretočen in v primeru, da ni, koliko izdelkov od tisoč ostane nepregledanih ob koncu simulacije . Poleg tega je potrebno ugotoviti tudi obremenjenost posameznega kontrolorja in število izdelkov, ki jih vsak izmed njih pregleda.

Blokovni diagram



Slika 45: Delni rezultati simulacije za model delavnice

Simulacijski model

```
         SIMULATE
*
*        Simulacija kontrole izdelkov
*        Osnovna časovna enota: 1 minuta
*
*        Eno delovno mesto s kapaciteto 3
*
*        Definiramo tabelo
*
TABELA   TABLE M1,5,5,10
*
*        Deklaracija kapacitet
*
         STORAGE      S(KON),3
*
*        GPSS/H blokovni del
*
*
         GENERATE     5                    Izdelki prihajajo iz avtomata vsakih 5 min
*
*        Prvi tekoči trak in kontrolor
*
         ADVANCE      2                    Tekoči trak 1
         QUEUE        VRSTA                Vstop v vrsto
         ENTER        KON                  Kontrolna enota prične z delom
         DEPART       VRSTA                Ven iz vrste
         ADVANCE      12,9                 Pregledovanje izdelka
         LEAVE        KON                  Izdelek zapusti delovno mesto prvega kont.
         TABULATE     TABELA               Tabela časov kontroliranja izdelkov
         TERMINATE    1
*
*        GPSS/H kontrolni stavki
*
         START 10,NP                       TG1 postavimo na 10; NP - brez izpisa
         RESET                             Resetiramo vso statistiko
         START 1000                        TG1 (terminations to go) postavimo na 1000
         END
```

Slika 46: Simulacijski model delavnice za eno vrsto pred kontrolorji

Izdelki prihajajo iz avtomata vsakih pet minut, kar modeliramo z blokom GENERATE 5. Zatem izdelki do vrste (QUEUE VRSTA) pred kontrolorji potujejo dve minuti (ADVANCE 2). Ko je eden izmed kontrolorjev prost (ENTER KON) vzame izdelek iz vrste (DEPART VRSTA) in ga prične pregledovati (ADVANCE 12,9). Po končanem pregledu, kontrolor izdelek vrne na tekoči trak (LEAVE KON). Z blokom TABULATE TABELA se tabelirajo časi kontroliranja izdelkov, za njim pa izdelki zapustijo sistem skozi blok TERMINATE 1.

Simulacija je izvedena najprej za 10 strank, kjer ukaz START 10,NP požene model za deset strank, kjer se rezultati ne izpišejo v statistiko. Zatem uporabimo kartico RESET, kjer resetiramo vso statistiko, transakcije, ki se nahajajo v modelu pa v njem ostanejo. Zatem simuliramo model za tisoč strank in zberemo rezultate.

Rezultati

STORAGE	TOTAL TIME (AVG-UTIL-DURING)	AVAIL TIME (AVG-UTIL-DURING)	UNAVL TIME (AVG-UTIL-DURING)	ENTRIES	AVERAGE TIME/UNIT	CURRENT STATUS	PERCENT AVAIL	CAPACITY	AVERAGE CONTENTS	CURRENT CONTENTS	MAXIMUM CONTENTS
KON	0.801			1001	12.001	AVAIL	100.0	3	2.402	1	3

QUEUE	MAXIMUM CONTENTS	AVERAGE CONTENTS	TOTAL ENTRIES	ZERO ENTRIES	PERCENT ZEROS	AVERAGE TIME/UNIT	$AVERAGE TIME/UNIT	QTABLE NUMBER	CURRENT CONTENTS
VRSTA	3	0.125	1000	725	72.5	0.623	2.266		0

TABLE TABELA

ENTRIES IN TABLE	MEAN ARGUMENT	STANDARD DEVIATION	SUM OF ARGUMENTS	
1000.0000	14.6461	5.4970	14646.1023	NON-WEIGHTED

UPPER LIMIT	OBSERVED FREQUENCY	PERCENT OF TOTAL	CUMULATIVE PERCENTAGE	CUMULATIVE REMAINDER	MULTIPLE OF MEAN	DEVIATION FROM MEAN
...						
10.0000	244.0000	24.4000	24.40	75.60	0.6828	-0.8452
15.0000	269.0000	26.9000	51.30	48.70	1.0242	0.0644
20.0000	280.0000	28.0000	79.30	20.70	1.3656	0.9740
25.0000	191.0000	19.1000	98.40	1.60	1.7069	1.8836
30.0000	14.0000	1.4000	99.80	0.20	2.0483	2.7932
35.0000	2.0000	0.2000	100.00	-0.00	2.3897	3.7028

Slika 47: Delni rezultati simulacije za model delavnice

Rezultati s Slika 47 prikazujejo, da so kontrolorji zasedeni 80% (KON 0.801). Najdaljša vrsta izdelkov pred kontrolorji so trije izdelki, kar ni praktično nobena omejitev za prikazani sistem. 725 izdelkov so kontrolorji prevzeli direktno v pregled (Zero entries), ostalih 275 pa je v povprečju čakalo 2.266 minute ($Average time/unit). Ob koncu simulacije je v pregledu še en izdelek(Current contents 1).

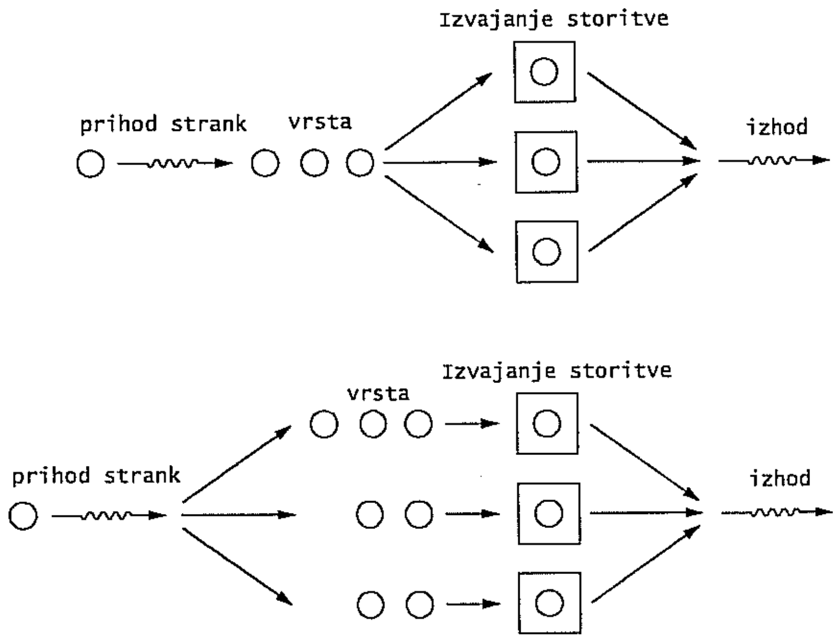
Spodnja tabela (Table Tabela) rezultatov na Slika 47 prikazuje tabelirane seštete čase čakanja izdelka na pregled in samega pregleda izdelka. Tabela je shranila rezultate za 1000 izdelkov (Entries in table), kjer je povprečje čakanja in pregleda 14,6461 minute (Mean argument), standardni odmik pa 5,4970 minute (Standard deviation).

Tako je očitno, da je prej kot v 10 minutah (Upper limit) pregledanih 244 (Observed frequency), med deset in petnajst minut je pregledano 269, med petnajst in dvajset minut je pregledanih 280, do petindvajset minut je pregledano 191, v več kot petindvajset minutah pa 16 (14+2) izdelkov. Poleg števila izdelkov v posameznem časovnem intervalu je prikazan tudi delež od celote (Percent of total), kumulativni delež (Cumulative percentage), kumulativni ostanek (Cumulative remainder), zmnožek povprečja (Multiple of Mean), ki je definiran kot: UPPER LIMIT /MEAN ARGUMENT in odmik od povprečja (Deviation of Mean).

Banka 2 in/ali Banka 3

Definicija modela

Prikazan bo model banke s tremi strežnimi mesti, kjer se izvajanje storitev lahko odvija na dva načina. Glede na prostorske omejitve in na način strežbe, ki ga želimo uvesti imamo v sistemu lahko eno vrsto za vse tri strežnike ali pa ima vsak strežnik svojo vrsto. Možna modela strežbe shematsko prikazuje Slika 48.



Slika 48: Možna modela strežbe v novi bančni poslovalnici.

Če pri vzpostavitvi načina strežbe ni prostorskih omejitev, želimo v banko uvesti sistem strežbe, ki zagotavlja pri podanih možnostih najhitrejši pretok strank skozi sistem in čim krajše čase čakanja na strežbo.

Problem

Pri izgradnji nove bančne podružnice, se vodstvo banke sooča z vprašanjem, kakšen sistem strežbe uvesti, da bo čas čakanja strank čim krajši. Od odločitev za sistem strežbe je odvisna tudi razporeditev delovnih mest in zasnova poslovalnice, zato je potrebno podati pravilno oceno delovanja strežbe za oba zgoraj prikazana načina.

Naloga

Za oba strežna sistema prikazana na Slika 48 zgradi model v GPSSu in izvedi simulacijski eksperiment za osem urni čas delovanja banke. Rezultate simulacije analiziraj in podaj poročilo o razliki med propustnostjo strežnih sistemov.

Kriteriji

- določi povprečni čas čakanja strank za oba zgoraj prikazana sistema
- ugotovi, ali je število strežnih mest zadostno glede na pričakovano količino strank nove banke

Simulacijski model

V vseh modelih s funkcijo pripišemo strankam število dokumentov, ki jih bančni uslužbenec obdela, da je posamezna stranka postrežena. Število dokumentov v modelu pripišemo z uporabo bloka ASSIGN in pomočjo diskretne funkcije FDOCS. Spremenljivko I uvedemo za potrebe DO zanke ker vsak simulacijski model ponovimo desetkrat po osem ur. S pomočjo stavka CLEAR pa odstranimo transakcije iz modela, hkrati pa ne vplivamo na generator naključnih števil. S tem se delovanje modela približa realnemu stanju.

Slika 49 prikazuje simulacijski model z eno vrsto in tremi strežnimi mesti, kjer so definirani trije enakovredni strežniki. Vse stranke se po prihodu v banko postavijo v skupno vrsto. Ko je prvi strežnik prost, gre najdlje čakajoči k prostemu strežnemu mestu. Po opravljeni storitvi stranka zapusti sistem.

```
*
*       Program Banka - 1 VRSTA, 3 STREŽNIKI
*       Verzija 1.1, 13.01.2000
*       prof. dr. Miroljub Kljajić, mag. Igor Bernik
*
*       Časi so podani v minutah
*
*       Deklaracije
FDOCS . FUNCTION RN(5),D5       Funkcija FDOCS, uporabljen naključni generator 5,
.3,1/.55,2/.75,3/.9,4/1,5       30% strank ima 1 dokument, 25% ima 2, 20% ima 3, 15% ima 4 in 10% jih ima 5
*
        STORAGE S(US),3         Imamo tri enakovredne strežnike
*
        INTEGER    &I           &I je spremenljivka, uporabljena za števec DO zanke
*
        SIMULATE                Izvajamo simulacijo
*
        GENERATE   .7,.65       Prihod strank v banko - v povprečju vsakih 42 sekund z odklonom 39 sekund
        ASSIGN     DOCS,FN$FDOCS,PH       Stranki pripišemo število dokumentov po funkciji FDOCS
*
        QUEUE      VR           Stranka stopi v vrsto
        ENTER      US           Prosto uslužbenko/ca stranka zasede za čas strežbe
        DEPART     VR           Ko stranka stopi k okencu izstopi iz vrste
        ADVANCE    .8*PH(DOCS)  Strežba stranke, za vsak dokument po 48 sekund
        LEAVE      US           Po opravljenih poslih odidemo in prepustimo uslužbenko/ca naslednjemu
*
        TERMINATE               Izhod - stranke odhajajo iz banke
*
*       GPSS kontrolni stavki
*
        GENERATE   480          Čas delovanja banke je 8 ur
        TERMINATE  1            Simuliramo za 1 dan
        DO         &I=1,10      Izvedemo simulacijo 10 krat
        START      1            Po en dan
        CLEAR                   Stranke, ki ostanejo po zaprtju banke se postrežejo in odidejo domov
        ENDDO                   Konec zanke DO
*
        END                     Sedaj pa je konec simulacije!
```

Slika 49: Program v jeziku GPSS za model banke z eno vrsto in tremi strežniki

Slika 50 prikazuje simulacijski model z tremi vrstami in tremi strežnimi mesti. Po prihodu stranke v banko se le ta mora odločiti za eno izmed vrst pred posameznim strežnim mestom. V spodnjem programu je izbira izvedena s pomočjo blokov TEST.

V prvem bloku (TEST LE Q$VR1,Q$VR2,ADR1) preverimo, ali je vrsta 1 krajša ali enaka kot vrsta 2. Če je pogoj izpolnjen – torej je odgovor DA, transakcija nadaljuje pot v naslednjo vrstico programa. V primeru negativnega odgovora transakcija skoči na segment z oznako ADR1.

V naslednji vrstici (TEST LE Q$VR1,Q$VR3,ADR3) preverimo, če je vrsta 2 krajša ali enaka kot vrsta 3. V primeru pritrdilnega odgovora transakcija nadaljuje pot skozi model, v nasprotnem pa skoči na segment z oznako ADR3.

Če pri prvem test stavku (TEST LE Q$VR1,Q$VR2,ADR1) dobimo negativen odgovor in transakcija skoči na segment ADR1, tam zopet preverjamo (TEST LE Q$VR2,Q$VR3,ADR3), ali je vrsta 2 krajša ali enaka kot vrsta 3. V primeru pozitivnega odgovora program transakcijo brez pogojno preusmeri (TRANSFER ,ADR2) na segment ADR2, pri negativnem odgovoru pa transakcija skoči na segment ADR3.

Vrsta in strežba pri vsakem strežniku poteka posebej, ob koncu opravljanja storitve pa stranka zapusti sistem s skokom na segment z oznako KONEC.

```
*
*       Program Banka - 3VRSTE, 3STREŽNIKI
*       Verzija 1.1, 13.01.2000
*       prof. dr. Miroljub Kljajić, mag. Igor Bernik
*
*       Časi so podani v minutah
*
*       Deklaracije
FDOCS  FUNCTION RN(5),D5        Funkcija FDOCS, uporabljen naključni generator 5,
.3,1/.55,2/.75,3/.9,4/1,5       30% strank ima 1 dokument, 25% ima 2, 20% ima 3, 15% ima 4 in 10% jih ima 5
*
       INTEGER     &I           &I je spremenljivka, uporabljena za števec DO zanke
*
       SIMULATE                   Izvajamo simulacijo
*
       GENERATE    .7,.65       Prihod strank v banko - v povprečju vsakih 42 sekund z odklonom 39 sekund
       ASSIGN      DOCS,FN$FDOCS,PH Stranki pripišemo število dokumentov po funkciji FDOCS
*
       TEST         LE Q$VR1,Q$VR2,ADR1    Je vrsta 1 krajša ali enaka kot vrsta 2
       TEST         LE Q$VR1,Q$VR3,ADR3    Je vrsta 2 krajša ali enaka kot vrsta 3
*
       QUEUE       VR1          Stranka se odloči za vrsto 1
       SEIZE       US1          Prosto uslužbenko/ca stranka zasede za čas strežbe
       DEPART      VR1          Ko stranka stopi k okencu izstopi iz vrste 1
       ADVANCE     .8*PH(DOCS)  Strežba stranke, za vsak dokument po 48 sekund
       RELEASE     US1          Po opravljenih poslih odidemo in prepustimo uslužbenko/ca naslednjemu
       TRANSFER    ,KONEC       Stranka se usmeri na izhod iz banke
*
ADR2   QUEUE       VR2          Stranka se odloči za vrsto 2
       SEIZE       US2          Prosto uslužbenko/ca stranka zasede za čas strežbe
       DEPART      VR2          Ko stranka stopi k okencu izstopi iz vrste 2
       ADVANCE     .8*PH(DOCS)  Strežba stranke, za vsak dokument po 48 sekund
       RELEASE     US2          Po opravljenih poslih odidemo in prepustimo uslužbenko/ca naslednjemu
       TRANSFER    ,KONEC       Stranka se usmeri na izhod iz banke
*
ADR1   TEST        LE Q$VR2,Q$VR3,ADR3     Je vrsta 2 krajša ali enaka kot vrsta 3
       TRANSFER    ,ADR2                   Če je zgornji pogoj izpolnjen se stranka odloči za vrsto 2
*
ADR3   QUEUE       VR3          Stranka se odloči za vrsto 3
       SEIZE       US3          Prosto uslužbenko/ca stranka zasede za čas strežbe
       DEPART      VR3          Ko stranka stopi k okencu izstopi iz vrste 3
       ADVANCE     .8*PH(DOCS)  Strežba stranke, za vsak dokument po 48 sekund
       RELEASE     US3          Po opravljenih poslih odidemo in prepustimo uslužbenko/ca naslednjemu
*
KONEC  TERMINATE                  Izhod - stranke odhajajo iz banke
*
*      GPSS kontrolni stavki
*
       GENERATE    480          Čas delovanja banke je 8 ur
       TERMINATE   1            Simuliramo za 1 dan
       DO          &I=1,10      Izvedemo simulacijo 10 krat
       START       1              Po en dan
       CLEAR                      Stranke, ki ostanejo po zaprtju banke se postrežejo in odidejo domov
       ENDDO                      Konec zanke DO
*
       END                      Sedaj pa je konec simulacije!
```

Slika 50: Program v jeziku GPSS za model banke s tremi vrstami in stavki TEST

Za podrobneje razumevanja delovanja programa si poskušajte pomagati z komentarji, ki so podani ob programu.

Funkcijsko enak programu s Slika 50 je program, ki ga prikazuje Slika 51, le ta smo v tem primeru uporabili blok SELECT. Pri tem stavku izbiramo (SELECT MIN 2,1,3,,Q) najkrajšo (MIN) vrsto, ki jo hrani parameter P2 (2), kjer je število vrst od 1 do 3 (1,3). Ker se stranka ne glede na dolžino vrst vedno v eno razvrsti, le te ne preverjamo s številom („), pač pa glede na trenutno dolžino vrst (Q).

Dolžina strežbe je odvisna od števila dokumentov, ki jih ima stranka, po opravljeni storitvi pa stranka zapusti sistem.

```
*
*       Program Banka - 3 VRSTE, 3 STREŽNIKI
*       Verzija 1.1, 13.01.2000
*       prof. dr. Miroljub Kljajić, mag. Igor Bernik
*
*       Časi so podani v minutah
*
*       Deklaracije
FDOCS   FUNCTION RN(5),D5       Funkcija FDOCS, uporabljen naključni generator 5,
.3,1/.55,2/.75,3/.9,4/1,5       30% strank ima 1 dokument, 25% ima 2, 20% ima 3, 15% ima 4 in 10% jih ima 5
*
        INTEGER    &I           &I je spremenljivka, uporabljena za števec DO zanke
*
        SIMULATE                 Izvajamo simulacijo
*
        GENERATE   .7,.65       Prihod strank v banko - v povprečju vsakih 42 sekund z odklonom 39 sekund
        ASSIGN     DOCS,FN$FDOCS,PH Stranki pripišemo število dokumentov po funkciji FDOCS
*
        SELECT MIN 2,1,3,,Q     Izberemo MIN-vrsto, 2-parameter v katerega zapišemo vrednost,
*                               1,3-od vrste 1 do 3, Q-primerjamo vrste
        QUEUE      P2           Stranka se odloči za najkrajšo vrsto (odvisna od parametra P2)
        SEIZE      P2           Prosto uslužbenko/ca stranka zasede za čas strežbe
        DEPART     P2           Ko stranka stopi k okencu izstopi iz vrste v kateri stoji
        ADVANCE    .8*PH(DOCS)  Strežba stranke, za vsak dokument po 48 sekund
        RELEASE    P2           Po opravljenih poslih odidemo in prepustimo uslužbenko/ca naslednjemu
        TERMINATE               Izhod - stranke odhajajo iz banke
*
*       GPSS kontrolni stavki
*
        GENERATE   480          Čas delovanja banke je 8 ur
        TERMINATE  1            Simuliramo za 1 dan
        DO         &I=1,10      Izvedemo simulacijo 10 krat
        START      1            Po en dan
        CLEAR                   Stranke, ki ostanejo po zaprtju banke se postrežejo in odidejo domov
        ENDDO                   Konec zanke DO
*
        END                     Sedaj pa je konec simulacije!
```

Slika 51: Program v jeziku GPSS za model banke s tremi vrstami in stavkom SELECT

Za podrobneje razumevanja delovanja programa si poskušajte pomagati z komentarji, ki so podani ob programu.

Rezultati

Rezultati pri vseh modelih so podani samo za prvi tek simulacije! Da dobimo prave rezultate, je potrebno posamezne vrednosti pri vseh desetih tekih vzeti kot povprečno vrednost in šele na podlagi teh vrednosti sklepati o dejanskih vrednostih!

STORAGE	--AVG-UTIL-DURING-- TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/UNIT	CURRENT STATUS	PERCENT AVAIL	CAPACITY	AVERAGE CONTENTS	CURRENT CONTENTS	MAXIMUM CONTENTS
US	0.941			687	1.972	AVAIL	100.0	3	2.822	2	3

QUEUE	MAXIMUM CONTENTS	AVERAGE CONTENTS	TOTAL ENTRIES	ZERO ENTRIES	PERCENT ZEROS	AVERAGE TIME/UNIT	$AVERAGE TIME/UNIT	QTABLE NUMBER	CURRENT CONTENTS
VR	16	3.268	687	117	17.0	2.283	2.752		0

Slika 52: Rezultati za model banke z eno vrsto in tremi strežniki

Iz rezultatov modela z eno vrsto in tremi strežnimi mesti (Slika 52) je razvidno, da so strežna mesta (STORAGE) v povprečju zasedena okoli 94%, v času simulacije pa je postreženo 687 strank, ob koncu simulacije pa sta se v strežbi nahajali še dve stranki.

V vrsti je bilo maksimalno 16 strank, v povprečju dobre tri stranke (3.268), skupaj je prišlo 687 strank, direktno v strežbo je šlo 117 strank, povprečni čas čakanja strank, ki so dejansko čakale pa je približno 2 minuti in 45 sekund (2.752).

```
Rezultati za banko s tremi vrstami in tremi strežnimi mesti pa so prikazani na Slika 53 in    QUEUE
MAXIMUM     AVERAGE     TOTAL       ZERO     PERCENT    AVERAGE    $AVERAGE    QTABLE    CURRENT
           CONTENTS   CONTENTS    ENTRIES    ENTRIES     ZEROS    TIME/UNIT   TIME/UNIT   NUMBER    CONTENTS
    VR1          6      1.745       252          8        3.2       3.324       3.432                 1
    VR2          5      1.437       238         20        8.4       2.897       3.163                 1
    VR3          5      1.192       197         28       14.2       2.905       3.386                 0
```

Slika 54. Kot je razvidno so enaki, kar potrjuje, da sta programa identična.

Vidimo, da so posamezni uslužbenci zasedeni različno, kar je odvisno od postavljanja strank v vrsto. Skupaj je v času simulacije postreženih 685 strank, dve pa sta ob koncu simulacije še v vrsti. Najdaljša je bila vrsta 1, ki je vsebovala največ šest strank, vrsti 2 in 3 pa po največ pet strank. Bolj zanimiv je povprečni čas čakanja strank, ki so dejansko čakale na strežbo, ki je v vseh treh vrstah precej daljši kot tri minute, v primerjavi z zgornjimi rezultati, kjer je povprečni čas čakanja krajši za približno pol minute.

```
          --AVG-UTIL-DURING--
FACILITY  TOTAL  AVAIL  UNAVL      ENTRIES     AVERAGE   CURRENT  PERCENT  SEIZING  PREEMPTING
           TIME   TIME   TIME                 TIME/XACT   STATUS   AVAIL     XACT       XACT
     US1  0.996                       251       1.905     AVAIL              682
     US2  0.966                       237       1.956     AVAIL              686
     US3  0.852                       197       2.075     AVAIL

   QUEUE     MAXIMUM     AVERAGE     TOTAL       ZERO    PERCENT    AVERAGE    $AVERAGE   QTABLE   CURRENT
            CONTENTS    CONTENTS    ENTRIES    ENTRIES    ZEROS    TIME/UNIT   TIME/UNIT  NUMBER   CONTENTS
     VR1           6       1.745       252          8       3.2      3.324       3.432                1
     VR2           5       1.437       238         20       8.4      2.897       3.163                1
     VR3           5       1.192       197         28      14.2      2.905       3.386                0
```

Slika 53: Rezultati za model banke s tremi vrstami in stavki TEST

```
          --AVG-UTIL-DURING--
FACILITY  TOTAL  AVAIL  UNAVL      ENTRIES     AVERAGE   CURRENT  PERCENT  SEIZING  PREEMPTING
           TIME   TIME   TIME                 TIME/XACT   STATUS   AVAIL     XACT       XACT
     US1  0.996                       251       1.905     AVAIL              682
     US2  0.966                       237       1.956     AVAIL              686
     US3  0.852                       197       2.075     AVAIL

   QUEUE     MAXIMUM     AVERAGE     TOTAL       ZERO    PERCENT    AVERAGE    $AVERAGE   QTABLE   CURRENT
            CONTENTS    CONTENTS    ENTRIES    ENTRIES    ZEROS    TIME/UNIT   TIME/UNIT  NUMBER   CONTENTS
     VR1           6       1.745       252          8       3.2      3.324       3.432                1
     VR2           5       1.437       238         20       8.4      2.897       3.163                1
     VR3           5       1.192       197         28      14.2      2.905       3.386                0
```

Slika 54: Program v jeziku GPSS za model banke s tremi vrstami in stavkom SELECT

Iz zgornjih rezultatov je razvidno, da je strežni sistem, ki ima eno vrsto in več strežnih mest bolj primeren od sistema, kjer ima vsak strežnik svojo vrsto. Ker se tudi pri nas vse bolj uveljavljajo veliki sistemi strežbe (hipermarketi, bančne poslovalnice, ...) menimo, da bi graditelji le teh morali bolj razmisliti o zasnovi vrst pred strežnimi mesti. Z eno vrsto ne le da skrajšamo čas čakanja in s tem povečamo zadovoljstvo strank, pač pa potrebujemo tudi manj prostora in zagotovimo večjo pretočnost sistema za strežbo strank.

8 Delo z GPSS/H-jem

GPSS/H je splošen simulacijski sistem, ki teče v okolju MS-DOS
Delovanje:
simulacijski model, ki je zapisan v ASCII datoteki s končnico .GPS (npr. Joebarb.gps) izvede, rezultate pa zapiše v datoteko s končnico .LIS (npr. Joebarb.lis)

Navodila za namestitev

Po prenosu datoteke gpssh.zip preko omrežja uporabite program za stiskanje podatkov WinZip.

Program poženete z ukazom gpssh (v dos-ovski vrstici oz. Command prompt - u, v mapi, kjer so gpss/h datoteke).
Kot test vpišite v dos-ovski vrstici: "gpssh joebarb". (brez narekovajev)
Ki vam bo izpisala rezultate v datoteko joebarb.lis

Model lahko napišemo v urejevalniku teksta Notepad ali Beležnici. Pri tem shranimo datoteko s končnico .GPS tako, da pri shranjevanju datoteke vpišemo ime med dvojna narekovaja.
Primer:

File/Save As/File Name: "Joebarb.gps"

9 Viri

Banks.J, Carson.S.J.II, Ngo J. S. : GETTING STARTED WITH GPSS/H , Virginia, 1989

Dijk J. N. in sodelavci: Visual Interactive Modelling with SimView for Organizational Improvement, Simulation, 67, No. 2, 1996, 106-120

FORRESTER, Jay, W.: Industrial Dynamics, The M.I.T. Press, 1973, str. 13, ISBN 0 262 56001 1

GORDON G: Sytem Simulation, Prentice-Hall, Inc. Englewood Cliffs, New Jersey

GORDON, G., System Simulation, Prentice-Hall, New Jersey, 1969

Harell C. R., Bateman R. E., Gogg T. J. Mott J. R.: System Improvement Using Simulation 4th ed., ProMODEL Corporation, Orem, 1996

Homer, B. J.: Why we iterate: Scientific modeling in theory and practice, System Dynamics Review, Vol. 12, 1, 1-19, John Wiley & Sons, 1996

Kljajić M. in sodelavci: Simulacijski sistem za podporo pri odločanju v poslovnih sistemu, financer Ministrstvo za znanost in tehnologijo, šifra J5-6218-0586, Fakulteta za organizacijske vede, Kranj, poročilo o delu, 1994-1996

KLJAJIĆ M., LESKOVAR R., (1994),: Multicriteria Assessment of Simulation Scenario for Business Decision Suport, IASTED, Applied Simulation & Modelling, Editor: M.H. Hamza, Acta Press, Anaheim-Calgary-Zurich

KLJAJIĆ M., LESKOVAR R., GRADIŠAR M., (1990),: Validation of Management Simulation Model and Its Parameter Sensitivity Analysis, IASTED, Applied Simulation & Modelling, Editor: M.H. Hamza, Acta Press, Anaheim-Calgary-Zurich

KLJAJIĆ M., LESKOVAR R., ŠKRABA A., RAJKOVIĆ V., BITENC I.,: Multicriteria evaluation of a simulation scenario for business decision support, Proceedings of the IASTED international conference on Modelling, Simulation and Optimization, Gold Coast, Australia, 19969.

Kljajić M., Teorija sistemov, Moderna organizacija – FOV Kranj, 1994

Kljajić, M., Leskovar, R., Škraba, A., Rajkovič, V., Bitenc, I., Multicriteria evaluation of simulation scenario for business decision support, V: Hamza, M. H. (ur.). Proceedings of the IASTED International Conference on Modelling, Simulation and Optimization, May 6-9, 1996, Gold Coast, Australia. Compact Disc digital data. [S. l.]: The International Association of Science and Technology for Development - IASTED, cop. 1996

KLJAJIĆ, M., RAJKOVIČ, V. JESENKO, J. GRADIŠAR, M., LESKOVAR, R., ČIŽMAN, A., BITENC, I., ŠKRABA, A. BERNIK, I., Simulacijski sistem za podporo pri odločanju v poslovnih sistemih : zaključno poročilo o rezultatih opravljenega znanstveno-raziskovalnega dela na področju temeljnega raziskovanja, Kranj : Fakulteta za organizacijske vede, 1997

Leskovar R., Kljajić M., Bernik I., Škraba A., Bitenc I.: Integracija programskih orodij pri razvoju in uporabi simulacijskega modela poslovnega sistema I. in II., Dnevi slovenske informatike, Zbornik posvetovanja, Slovensko društvo informatika, Portorož, 1997

LESKOVAR R.: Metode večkriterijske izbire scenarija simulacijskega modela za podporo pri odločanju v poslovnih sistemih, doktorska disertacija, Fakulteta za organizacijske vede Kranj, 1993

Özdemirel N. E., Yurttas G. Y., Köksal G.: Computer-Aided Planning and Design of Manufacturing Simulation Experiments, Simulation, 67, No. 3, 1996, 171-191

Rajković V., J. Efstathiou and M. Bohanec (1987). A Concept of Rule-Based Decision Support Systems. Optimization Models Using Fuzzy Sets and Possibility Theory (J. Kacprzyk, J. Orlovski and D. Reidel, Ed.). A.S., Publishing Company, Dordrecht.

SAATY T.L.: Multicriteria Decision Making; The Analytic Hierarchy Process, RWS Publications, Pittsburg, 1990

Saltman R. M.: An Animated Simulation Model for Analyzing On-Street Parking Issues, Simulation, 69, No. 2, 1997, 79-90

Schriber Thomas J. : AN INTRODUCTION TO SIMULATION USING GPSS/H, 1991

Schriber Thomas J. : SIMULATION USING GPSS, 1974

Tung, X. B., Co-oP A group decision Support System for Cooperative Multiple Criteria Group Decision Making, Springer – Verlag, 1987

Žiljak, Vilko, Simulacija računalom, Zagreb : Školska knjiga, 1982

10 Navodilo za izdelavo seminarske naloge

Vaje pri predmetu usposobijo slušatelja za izdelavo seminarske naloge. Slušatelj sam ali po dogovoru s profesorjem izbere temo seminarske naloge, ki jo zagovarja.

Pri večini simulacijskih projektov poteka delo v skupini, zato je potrebno zagotoviti natančno dokumentacijo in primerno predstavitev rezultatov. Dokumentacija simulacijskega projekta naj bi glede modela in simulacije vsebovala:

1. Neformalni opis modela in utemeljitev predpostavk
2. Situacijska shema oz. vzročno – posledični diagram
3. Formalni opis – blokovni diagram (za zvezne in za diskretne modele) in matematični model (za zvezne modele podati sistem enačb pri diskretnih modelih podati časovne porazdelitve. Analizirati čase med prihodi, čase strežbe, razmerja med spremenljivkami v modelu (zveznem), v primeru, da vzorčite čase (diskretni model) prikažite histogram, v primeru, da imate zgodovinske podatke, npr. rast prebivalstva v državi prikažite funkcijo rasti (zvezni model))
4. Simulacijski program (podate ga v prilogi, na predstavitvi predstavite samo morebitne posebnosti v modelu)
5. Analiza rezultatov simulacijskih eksperimentov (rezultate obvezno predstaviti tabelarično in grafično)
6. Povezava predpostavljenega modela s podobnimi
7. Določitev uporabnosti rezultatov ter analiza stroškov/koristi, podajte kriterijsko funkcijo
8. Sklepne ugotovitve

Pri diskretnih modelih zaradi naključnih generatorjev izvedite več simulacijskih tekov ter statistično obdelajte rezultate (CLEAR, RESET). Podajte enoto simulacijske ure ter opredelite za katero časovno obdobje izvajate simulacijo (konica, normalno delovanje, mrtvi tek). Pri zveznih modelih izvedite več simulacijskih scenarijev ter v obeh primerih interpretirajte rezultate.

V besedilu seminarske naloge podajte ali je problem realen ali hipotetičen.

Trajanje zagovora je omejeno na pet (5) minut z dodatnimi petimi minutami za diskusijo. Vsaka skupina naj za zagovor pripravi (okvirno) pet (5) prosojnic, z naslednjo strukturo:

1. podatki o seminarski nalogi (po priloženi predlogi - naslov, člani skupine itd.)
2. predstavitev problema in neformalna shema sistema
3. predstavitev zastavljenih ciljev
4. predstavitev rezultatov
5. zaključek

Na predstavitvi naj vsak član skupine predstavi določen del seminarske naloge.

Upoštevajte, da " 1 slika = 1000 besed ".

Prosojnice morajo biti napisane z velikostjo črk, najmanj 20 pt. in naj bodo tiskane ležeče. Na posamezni prosojnici naj bodo podana le dejstva.

Seminarska naloga se predstavi na zagovoru, dopolni ter nato odda na izpitu. Seminarsko nalogo oddate le v enem izvodu na papirju ter na disketi (model, besedilo seminarske naloge, podatki – vse, kar je bilo potrebno za izvedbo seminarske naloge, na disketi naj bodo napisani priimki članov skupine).

Delo je nastalo v okviru projekta Phare:

European Training Foundation
The Phare Multicountry program in Distance Education
etf/97/vet/0056, II/5-40/1998