Definiraj pojme "Vmesnik", "Programski uporabniški vmesnik", "Metafora delovne mize" in "Look and feel".

Vmesnik je mesto kjer se srečujejo in delujejo neodvisni sistemi, ki med seboj komunicirajo.
Programski uporabniški vmesnik združuje vhodne in izhodne naprave ter programsko opremo, te vire uporablja in podpira. Obsega: dokumentacijo; uvajanje; podporo in vse kar oblikuje izkušnje uporabnikov z računalnikom.
Metafora delovne mize predstavlja perspektivo, ko uporabnik vidi pred seboj okna kot liste papirja na mizi, katera lahko polaga enega preko drugega. Lahko pa tudi postavi oba drugega ob drugega, da vidi oboje podatke hkrati.
Look and feel grafičnega uporabniškega vmesnika vključuje njegovo oblikovanje (design), vključno z barvami, oblikami, načrti(layout) in črkovnimi stili(typefaces) - "look", kot tudi obnašanje dinamičnih elementov kot so gumbi, okvirji in menuji - "feel".

Naštej osnovne principe za gradnjo (faze načrtovanja) uporabniških vmesnikov in za vsakega vsaj en konkreten primer.

- zagotovi nadzor uporabnika (umogoči uporabo tipkovnice in miške) – fleksibilnost
    - prekinitev danih opravil – prekinljivost
- reduciraj obremenitev uporabnikovega spomina (zagotovi vizualne namige) – informiranost
    - zagotovi bljižnice – hitrost
- zagotovi konsistenčnost vmesnika (ohranjaj kontekst uporabnikovih opravil) – zveznost
    - zagotovi estetsko privlačnost in polnost - izgled.

Naštej osn. principe načrt. upor. vmesnikov ter naštej razlike med principi, navodili in standardi za načrtovanje vmesnikov.

- poln nadzor (ubijanje procesov)
- razbremenjevanje spomina (meni, ikona)
- konsistenčnost (uporaba enakih barv)

Principi so visokonivojski koncepti in navodila za načrtovanje GUV, nastali na podlagi človekovega mentalnega modela in njegovih fizičnih in psiholoških lastnosti.
Navodila lahko nastanejo šele po določenem času uporabe, se izpopolnjujejo(so sugestije). Za načrtovanje UV se nanašajo na: Predstavitev grafičnih gradnikov vmesnika uporabniku in estetiko, interakcijo uporabnika prek naprav in grafičnih gradnikov za interakcijo, Lastnosti gradnikov in njihove medsebojne relacije.
Standardi definirajo karakteristike objektov in sistemov, ki jih uporabljamo vsakodnevno, jih moramo upoštevati. Omogočajo komunikacijo načrtovalca s katerimkoli drugim načrtovalcem. Se ažurirajo, sicer zavirajo razvoj. Zakaj standardi? Lažja izbira in uporaba informacijske procesne opreme, eliminacija nekonsistenčnosti in razlik.

Kaj sta modela FEVH in WOSH? Kje zasledite enega in kje drugega? Obrazložite razlike med modeloma.

Zasledimo jih pri upor. vmesnikih.
WOSH modelu sledijo OUUV. W pomeni celo okno vmesnika, prva vrstica pa ima menije [Object(ime objekta), Selected(opcije nad izbranim delom), View, Help] kateri sledijo tipu objekta. Tak model najdemo v OS2.
FEVH model je podmodel WOSH modea. Če sledimo FEVH modelu je prva vrstica z meniji: [File, Edit, View,.., Help]. Uporabljajo ga GUV. FEVH model najdemo v OS Windows.
RAZLIKE:

- G: Uporabnik požene aplikacijo preden dela z njo
- O: Upor. odpre objekte v poglede preden dela z njimi
~ G: Ikone predstavljajo aplikacije
~ O: Ikone predst. objekte, ki jih lahko direktno manipuliramo
- G: Ena aplikacija je eno opravilo
- O: En objekt lahko uporabimo za različna opravila
~ G: Aplikacijo sestavljajo ikona, primarno ter sekundarna okna
~ O: Vmesnik sestavlja množica sodelujočih objektov in pogledov na njih
- G: Vsebina je prikazana s tekstovnimi listami in dialogi
- O: Vsebina je prikazana z vsebovalniki (mape – folder, in beležke - notebook)

S stališča vhodnega ("feel") in izhodnega ("look") modela načrtovanja uporabniških vmesnikov primerjajte med seboj naslednje gradnike uporabniških vmesnikov (oznaka, menu, podoba lista, podoba text).

- oznaka, menu ter podoba lista po številu opcij(v podobo lista lahko dodajamo nove opcije)
- menu ter podoba lista lahko primerjamo po dinamičnosti(podoba lista bolj dinamična?)
- podoba lista ter podoba tekst po manipulaciji nad tekstom()

Razložite razlike med konceptom akcija-objekt in konceptom objekt-akcija pri uporabniških vmesnikih. Naštejte tipične programske uporabniške vmesnike, ki temeljijo na enem ali drugem konceptu.

Akcija-objekt je koncpet, ko najprej napišemo akcijo, potem se le-ta izvede nad objektom. Primer je ukazna vrstica v okoljih DOS, bash (Unix). Objekt-akcija pa je koncept, ko najprej poiščemo objekt, potem pa izvedemo akcijo nad njim. To je tipično za operacijske sisteme. Primeri so Windows, X (Unix).

Razložite koncept direktne manipulacije. Kje ga uporabljamo?

Za direktno manipulacijo je značilno:
- vizualizacija vseh objektov (omizje, ikone – datoteke, direktoriji, programi), sistem je podaljšek realnega sveta;
- hitre, reverzibilne akcije;
- zvezna vidljivost objektov in akcij;
- prepoznavanje (razbremenjevanje hitrega spomina).

Slabosti so:
- težko načrtovanje, zahtevno programiranje, zmogljiva strojna oprema;
- ikone niso vedno razumljive, vse naloge se ne dajo opisati s konkretnimi objekti in direktno akcijo;
- delovna domena je sedanjost.

Prednosti so:
- hitro učenje;
- uporabnik neposredno vidi, če akcije vodijo k cilju;
- uporabnik pridobi samozaupanje in kontrolo

Razložite arhitekturo objektno usmerjenih uporabnišlih vmesnikov.

Objektno usmerjen uporabniški vmesnik (OUUV) je vmesnik, pri katerem se uporabnik ukvarja z objekti, ki predstavljajo realni svet in ni obremenjen z arhitekturo računalnika oz. kako uporabljati aplikacije in objekte. Lastnosti OUUV so:
- je simulacija realnosti oziroma realizacija objektov realnega sveta;
- vmesnik ponuja množico objektov;
- uporabnik manipulira z objekti;
- okno je pogled v objekt;
- ikona je objekt;
- klik po ikoni pomeni odpreti objekt.

Razložite mere zmogljivosti in subjektivna merila pri testiranju uporabnosti grafičnih vmesnikov.

Mere zmogljivosti(kvantitativne meje)
- število nalog, čas opravljanja nalog, napake, pomoč.

Subjektivna merila(kvalitativna merila)
- ocene glede na razpoznavanje, mnenja, želje in zadovoljstvo.

Razložite vlogo in način uporabe metafor pri načrtovanju uporabniških vmesnikov.

Metafora je prenašanje lastnosti objektov ali akcij realnega sveta na druge objekte z namenom sugeriranja podobnosti ali analogije med njimi. Ko se uporabnik in razvijalec dogovorita, se dogovorita tudi za metafore, ki pripomorejo k učinkovitejši uporabi sistema. Ikone so slike - metafore, ki opisujejo in predstavljajo objekte in akcije.

Razložite koncept odjemalec-strežnik pri okenskem sistemu X. Sedite za računalnikom z imenom alfa in poganjate grafično aplikacijo na računalniku beta. Kje je odjemalec in kje strežnik?

Strežnik kontrolira prikazovalnik: nadzoruje dostop do prikazovalnika za odjemalce, pošilja sporočila po mreži, sledi zahtevam odjemalcev in ažurira okna, dvodimenzionalno risanje, omogoča distribuirano procesiranje, multiprocesiranje, sprejema vhodne dogodke in jih posreduje odjemalcem. Odjemalci imajo dostop do prikazovalnika: tečejo simultano (več oken istočasno, editiranje več datotek naenkrat, poganjanje aplikacij na več računalnikih istočasno, aplikacije so neodvisne), uporabljajo mrežni protokol preko programskega vmesnika xlib, med seboj komunicirajo preko strežnika (delijo si podatke), vsak posreduje zahtevo strežniku za kreiranje vsaj enega okna, ki je naslednik osnovnega okna - zaslon (root window). Prikazovalnik (display) je abstrakcija, ki predstavlja vhodne in izhodne naprave (zaslon, tipkovnica, miška), enega uporabnika. Grafični strežnik je delovna postaja alfa, odjemalec pa delovna postaja beta.

Naštejte vse vrste dialogov pri sistemu Motif, ki jih poznate. Na kratko razložite njihove lastnosti.

Dialogi za obvestila vsebujejo kratko obvestilo; imajo gumbe OK, Cancel, Help; Namenjeni takojšnji uporabi kateri sledi razkroj. (ErrorDialog, InformationDialog, MessageDialog, QuestionDialog, VVarningDialog, ...)

Dialogi za izbiro prikazujejo listo alternativ. Imajo gumbe [OK, Apply, Cancel, Help]. Namenjeni periodični uporabi. (SelectionDialog, PromptDialog, FileSelectionDialog, CommandDialog)

Kaj je modalnost in kakšne modalne načine poznamo?

Modalnost je način zaklepanja oken, v primeru, ko se moramo pred nadaljno uporabo posvetiti dialogu. Če hočemo delati s programom se moramo posvetiti najprej modalnemu oknu. Dokler tega ne storimo se aplikacija ne odziva. Načini:
- nemodalni način (vsa okna lahko še naprej prosto uporabljamo),
- začetni modalni način (onemogoči uporabo starševskega okna),
- polni modalni način (onemogoči uporabo celotne aplikacije),
- sistemski modalni način (onemogoči uporabo celotnega sistema).

Naštejte vse vrste menujev pri sistemu Motif, ki jih poznate. Kdaj se pri načrt. danega vmesnika odločimo za vsakega od njih?

- vrstični menu (MenuBar) – kadar imamo več skupnih opcij
- izvlečni menu (PulldownMenu) – če je dovolj prostora ter skupne opcije
- dvižni menu (PopupMenu) – ko imamo hitro delo in malo prostora
- kaskadni menu (CascadingMenu) – pri medsebojnem izključevanju, iz ene opcije več opcij
- opcijski menu (OptionMenu) – če je ena opcija bolj verjetna jo izpostavimo

Razložite razlike med gradnjo, upravljanjem in realizacijo podob pri sistemu Motif

GRADNJA – gradimo(kreiramo) objekte in določimo izgled strukture
UPRAVLJANJE – kako so podobe razvrščene, vidnost(ko s pod. upravljamo) in nevidnost(ko s pod. ne upravljamo) podob
REALIZACIJA – predaja nadzora strežniku, ki realizira prikaz na zaslonu

Razložite argumente odzivnih funkcij pri sistemu Motif. Razložite strukturo XmDrawingAreaCallbackStruct. Kje se pojavlja? Kdaj jo uporabljamo? Kaj so komponente te strukture?

Določene so v kodi aplikacije (callback resources), uporabnik jih začuti.
Primer: XtAddCallback(button, XmNactivateCallback, button_pushed, NULL);
button – podoba za katero je instalirana odzivna funkcija
XmNactivateCallback – ime odzivnega resursa (leva tipka ali presledek)
button_pushed – kazalec na odzivno funkcijo
NULL – podatki za odzivno funkcijo
XmDrawingAreaCallbackStruct – pojavlja se pri podobi za risanje (DrawingArea), pri odzivni funkciji risalne površine
Komponente te strukture: int cursor; XEvent *event; Window window;

Razložite mehanizme delovanja notranjih funkcij in odzivnih funkcij (dogodki, signali) okolja GTK+ ter razložite njihove medsebojne interakcije. (20 %)

Oboji so pridruženi podobam, signali ne vračajo ničesar, dogodki vrnejo logično vrednost.
Npr.: Odzivna funkcija za delo z dogodki: gboolean EventHandler(GtkWidget *widget, GdkEvent *event, gpointer data) kjer se napodlagi switch (event-type) stavka zgodi določen dogodek
Npr.: Odzivna funkcija za signal(pritisk gumba na miški): void ButtonClicked(GtkButton *button, gpointer data) kjer se izvede določena procedura, definirana v funkciji
Primer delovanja: ko pritisnemo gumb se izvede določen dogodek, izvede se tudi notranja funkcija, ki poskrbi, da se gumb “umakne“ v notranjost (je pritisnjen).

Kaj so dogodki in signali v GTK+?

Signali so pridruženi podobam in ne vrnejo ničesar. Vrste signalov: destroy, draw, move_cursor, clicked, activate, select_all
Dogodki so pridruženi podobam in vrnejo logični tip. V imenu imajo večinoma še pripono event.

Naštej komponente glavnega okna aplikacije kot jih predlagajo navodila za načrtovanje grafičnih uporabniških vmesnikov in razloži njihove funkcije.

- okvir (meja okna);
- naslov (uporabnik lahko takoj prepozna kateri aplikaciji pripada in kakšen namen ima okno);
- sistemski menu (osnovne sistemske funkcije za delo z oknom – zapri, minimiziraj,..);
- gumbi za velikost okna (hiter dostop do manipulacij za velikost delovnega okna);
- vrstični menu (hirarhično nanizane funkcije aplikacije);
- delovno področje;
- vrstica za izpis statusa (pogled nad delovanjem aplikacije);
- vrstica za obvestila (pogled nad delovanjem aplikacije);
- vrstica za ukaze (nekatere ukaze lahko bistveno hitreje izvršimo, če jih preprosto odtipkamo);
- drsniki (enostavno premikanje po oknu);
- pregrade (ločevanje vsebinsko različnih pojmov);
- polje za prikaz permanentnih ukazov in opcij (hitrejši dostop do ukazov)

Razloži delovanje senzornega, kratkotrajnega in dolgotrajnega spomina pri človeku ter prenašanje informacij med njimi.

- senzorni spomin (SM) sprejema ogromno količino informacij, ki se jih niti ne zavedamo;
- kratkotrajni spomin (STM) hrani okoli 7 dejstev približno 30 sekund;
- dolgotrajni spomin (LTM) – v njem so informacije kodirane. Hrani lahko do 100.000 besed, vendar lahko dostopamo samo do 2000-3000.

Prenašanje informacij: informacija pride preko čutil in procesorjev za zaznavanje v SM, od tu pa v STM. Tu se lahko informacija na podlagi podatkov iz LTM preoblikuje s posredovanjem procesorjev za razumevanje. Ko dostopamo do podatkov v LTM se morajo le-ti dekodirati v procesorjih za razumevanje in prenesti v STM. Tu lahko dostopamo do njih.

Kateremu principu načrtovanja uporabniških vmesnikov sledijo menuji? Naštejte vse vrste menujev, ki jih poznate in jih med seboj primerjajte. (20 %)

Naštej in razloži bistvene razlike med grafičnimi uporabniškimi vmesniki in objektno usmerjenimi uporabniškimi vmesniki.

	grafični uporabniški vmesniki	objektno usmerjeni uporabniški vmesniki
aplikacija/vmesnik	aplikacija=ikona, primarno, sekundarna okna	vmesnik=množica sodelujočih objektov in pogledov na njih
ikona	ikone predstavljajo aplikacije	ikone predstavljajo objekte, ki jih lahko direktno manipuliramo
delo	poženemo aplikacijo pred delom z njo	odpremo pogled v objekt pred delom z njim
pozornost	pozornost nad osnovno nalogo kot je ta določena z aplikacijo	pozornost nad vhodom in izhodom za objekte ter nalogami
učenje	koncentrirano na aplikacijo in njene funkcije	koncentrirano na skupne pristope in izgled objektov
vsebina	prikazana s tekstovnimi listami ali dialogi	prikazana z vsebovalniki (folder, notebook)
ena	ena aplikacija je eno opravilo	en objekt lahko uporabimo za različna opravila

Naštej prednosti grafičnega prikaza informacij pri uporabniških vmesnikih.

Ker človek sprejema približno 80% informacij vizualno (močan slikovni spomin) so prednosti:
- hitrejše razpoznavanje;
- hitrejše učenje;
- hitrejša uporaba orodij;
- minimizacija abstraktrnega razmišljanja (ni ukazov s kompliciranо sintakso);
- manj napak;
- takojšnja povratna informacija;
- predvidljiv odziv;
- ni jezikovnih ovir;
- atraktivnost in manj stresov.

Kaj sta to modela FEVH in WOSH? Kje zasledite enega in kje drugega? Razložite razlike med obema modeloma. (20 %)

WOSH modelu sledijo objektno usmerjeni UV. Prva vrstica ima menije [Object, Selected itd...] kateri sledijo tipu objekta. Tak model najdemo v os2. FEVH model je podmodel WOSH modea. Če sledimo FEVH modelu je prva vrstica z meniji: [File, Edit, User .... Help]. Uporabljajo ga grafično usmerjeni UV. FEVH model najdemo v OS Windows.

Razložite modalnost dialogov! Naštejte vse vrste dialogov pri sistemu Motif, ki jih poznate. (20 %)

Modalnost je način zaklepanja oken, v primeru, ko se moramo pred nadaljno uporabo posvetiti dialogu. Če hočemo delati s programom se moramo posvetiti najprej modalnemu oknu. Dokler tega ne storimo se aplikacija ne odziva. Načini: Nemodalni način (vsa okna lahko še naprej prosto uporabljamo), začetni modalni način (onemogoči uporabo starševskega okna), polni modalni način (onemogoči uporabo celotne aplikacije), sistemski modalni način (onemogoči uporabo celotnega sistema).

Dialogi za obvestila

vsebujejo kratko obvestilo; imajo gumbe OK, Cancel, Help; Namenjeni takojšnji uporabi kateri sledi razkroj. (ErrorDialog, InformationDialog, MessageDialog, QuestionDialog, WarningDialog, ...)

Dialogi za izbiro

prikazujejo listo alternativ. Imajo gumbe [OK, Apply, Cancel, Help]. Namenjeni periodični uporabi. (SelectionDialog, PromptDialog, FileSelectionDialog, CommandDialog)

Naštej faze načrtovanja grafičnih uporabniških vmesnikov.

- razumevanje uporabnika
- strojna oprema in grafične sposobnosti sistema
- razumevanje zeljene aplikacije
- načrtovanje oken
- načrtovanje menujev
- interakcija uporabnika s sistemom
- interakcija uporabnika s sistemom preko oken
- aranžiranje gr. gradnikov za komunikacijo
- izbor barv
- izbor in načrtovanje ikon
- povratna informacija: obvestila, dialogi, navodila, pomoč, testiranje

2. Dana je aplikacija napisana v okolju **Motif**. Dogodke obravnavajo Xt notranje funkcije, podobe in odzivne funkcije pridružene podobam. Nariši in razloži diagram poteka dela z dogodki.

aplikacija
zanka dogodkov
v kateri podobi je prišlo do dogodka?
Q1 – ali je dogodek registriran v prevajalni tabeli za tao podobo
Q2 – ali je dogodku pridružena odzivna procedura
izvrši funkcijo, pridruženo temu dogodku za to podobo
Q1
Q2
odzivna funkcija

Aplikacija se nahaja v zanki dogodkov. Ko se kakšen dogodek zgodi, je treba najprej ugotoviti v kateri podobi je prislo do dogodka. Če ta dogodek ni registriran v prevajalni tabeli dogodkov za to podobo (recimo klik na label) potem se aplikacija vrne v zanko dogodkov. V nasprotnem primeru se izvrši funkcija, pridružena temu dogodku (klik na gumb – sprememba senčenja objekta), preveriti je treba le še ali je temu dogodku pridružena kakšna odzivna procedura. Če je, se le-ta izvrši (npr. odpre se novo okno, pojavi dialog...) in se aplikacija vrne v zanko dogodkov, sicer se aplikacija vrne v zanko dogodkov.

I. Naštej komponente glavnega okna aplikacije kot jih predlagajo navodila za načrtovanje grafičnih uporabniških vmesnikov in razloži njihove funkcije.

Komponente oken:

- okvir (meja okna)
- naslov (uporabnik lahko takoj prepozna kateri aplikaciji pripada in kakšen namen ima okno)
- sistemski menu (osnovne sistemske funkcije za delo z oknom – zapri, minimiziraj,..)
- gumbi za velikost okna (hiter dostop do manipulacij za velikost delovnega okna)
- vrstični menu (hirarhično nanizane funkcije aplikacije)
- delovno področje
- vrstica za izpis statusa (pogled nad delovanjem aplikacije)
- vrstica za obvestila (pogled nad delovanjem aplikacije)
- vrstica za ukaze (nekater ukaze lahko bistveno hitreje izvršimo, če jih preprosto odtipkamo)
- drsniki (enostavno premikanje po oknu)
- pregrade (ločevanje vsebinsko različnih pojmov)
- polje za prikaz permanentnih ukazov in opcij (hitrejši dostop do ukazov)

2. Razloži koncept direktne manipulacije pri objektno usmerjenih uporabniških vmesnikih.

Za direktno manipulacijo je značilno:

- vizualizacija vseh objektov (omizje, ikone – datoteke, direktoriji, programi) sistem je podaljšek realnega sveta
- hitre, reverzibilne akcije
- zvezna vidljiost objektov in akcij
- prepoznavanje (razbremenjevanje hitrega spomina)

Slabosti:

- težko načrtovanje, zahtevno programiranje, zmogljiva strojna oprema
- ikone niso vedno razumljive, vse naloge se ne dajo opisati s konkretnimi objekti in direktno akcijo
- delovna domena je sedanjost

Prednosti:

- hitro učenje
- uporabnik neposredno vidi, če akcije vodijo k cilju
- uporabnik pridobi samozaupanje in kontrolo

3. Sediš za računalnikom z imenom miki. Napiši in razloži vse potrebne sistemske ukaze za delo z X grafično aplikacijo, ki je na oddaljenem računalniku z imenom pluton. Na katerem od obeh računalnikov je strežnik in na katerem odjemalec ?

Na lokalnem racunalniku omogočim exportanje prikazovalnika: miki> xhost +pluton, potem naredim telnet ali remote login na oddaljeni racunalnik: telenet pluton ali rlogin pluton in tam izvozim prikazovalnik: export DYSPLAY=miki:0.

Glede na to, da je po terminologiji odjemalec aplikacija, ki teče na oddaljenem stroju na mreži je pluton odjemalec; strežnik pa je programski proces, ki komunicira z odjemalci X in jim daje potrebne vire za tekoče delo in potemtakem je to miki.

#2 Question - http://fri-info.net/forum/viewtopic.php?f=13&t=12267&st=0&sk=t&sd=a&start=210

Na vpraŠanje:
Razložite mehanizme delovanja notranjih funkcij in odzivnih funkcij (dogodki, signali) okolja GTK+ ter razložite njihove medsebojne interakcije.

sm jst napisu:

Oboji so pridruženi podobam, signali ne vračajo ničesar, dogodki vrnejo logično vrednost.
Npr.: Odzivna funkcija za delo z dogodki: gboolean EventHandler(GtkWidget *widget, GdkEvent *event, gpointer data) kjer se na podlagi switch (event-type) stavka zgodi določen dogodek
Npr.: Odzivna funkcija za signal(pritisk gumba na miŠki): void ButtonClicked(GtkButton *button, gpointer data) kjer se izvede določena procedura, definirana v funkciji
Primer delovanja: ko pritisnemo gumb se izvede določen dogodek, izvede se tudi notranja funkcija, ki poskrbi, da se gumb "umakne" v notranjost (je pritisnjen).