SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

KORISNIČKO SUČELJE SUSTAVA ZA KONSTRUIRANJE MEHANIČKIH SKLOPOVA

Magistarski rad

Nenad Bojčetić, dipl. inž.

Mentor:
Dr. sc. Dorian Marjanović, dipl. inž.

Zagreb, 1996

PREDGOVOR

Izrada ovog rada potaknuta je višegodišnjim iskustvom autora u primjeni i razvoju CAD aplikacija, te istraživanjima provedenim na razvoju korisnièkog suèelja. Rad je dio istraživanja na projektu broj 2-08-173 “Model inteligentnog CAE sustava” u kojem se razmatraju metode razvoja raèunalnih sustava za podršku svim fazama razvoja proizvoda. Korisnièko suèelje povezuje dijelove programskog sustava u kompaktnu cjelinu, te ostvaruje komunikaciju između raèunala i korisnika. Pri tome korisnièko suèelje omoguæuje kontrolu tijeka rada programskog sustava i provođenja procesa konstruiranja uz moguænost razlièitog prikaza svih podataka vezanih za odvijanje konstrucijskog procesa i prikaze stanja procesa definiranih scenarijem.

ZAHVALA

Ovim putem želio bih izraziti zahvalnost pokojnom profesoru Aurelu Kosteliæu pod èijim voditeljstvom je zapoèeta izrada ovog rada.

Posebno bih se zahvalio doc. dr. Dorianu Marjanoviću na preuzimanju voditeljstva rada, te korisnim savjetima i podršci tijekom izrade rada.

Također zahvaljujem djelatnicima Laboratorija za osnove konstruiranja na brojnim korisnim raspravama i svesrdnoj podršci u tijeku zajednièkog istraživaèkog rada.

Nenad Bojèetiæ

SAŽETAK

U radu je izložen model korisnièkog suèelja sustava za konstruiranje mehanièkih sklopova temeljem planova. Da bi se odredili zahtijevi i smjernice razvoja modela korisnièkog suèelja razmatran je proces konstruiranja, te je analiziran prikaz procesa konstruiranja planom.

Opisan je model "inteligentnog" CAE sustava èiji je sastavni dio razmatrano korisnièko suèelje. Temeljem provedenih analiza naznaèeni su zadaci i uloga korisnièkog suèelja u CAE sustavu. Razmatrana je trenutna situacija u razvoju grafièkih korisnièkih suèelja, te metode i modeli pristupa koncipiranju i kreiranju korisnièkih suèelja.

Prikazana je struktura modela korisnièkog suèelja uz opis pojedinih dijelova i programskih metoda korištenih u razvoju modela. Na osnovu opisa modela korisnièkog suèelja kreiran je prototip koji je implementiran ostvarujuæi neposrednu vezu s bazom scenarija.

Kljuène rijeèi:

konstruiranje strojeva, teorija konstruiranja, konstruiranje pomoæu raèunala, korisnièka suèelja

UDK 658.512.2:681.3:621

THE MODEL OF THE USER INTERFACE FORSYSTEM FOR MECHANICAL COMPONENTS DESIGN

SUMMARY

In this MS thesis a model of user interface for plan based design of mechanical assemblies is proposed.

The research directions of the user interface development are investigated, especially considering design process requirements. State of the art, models and approaches in the development of the graphic user interfaces are considered. The methods of the plan representation of design process are discussed. The Model of “intelligent” CAE system is also described. Role of the user interface in such CAE system is explained and the main parts of the user interface are determined.

The structure and the parts of the model of the user interface are presented and programming methods used in the development of the model are described. The model of designer oriented user interface is proposed and a prototype of the model is developed. The prototype of the user interface model interacts directly with scenario database.

Keywords:

mechanical design, design theory, computer aided design, user interface design

UDC 658.512.2:681.3:621

Sadržaj

1. Popis slika:

2. *Popis tablica:*

Sadržaj

Popis slika:

Popis tablica:

3. Uvod

U današnje vrijeme konstruktor se u svom svakodnevnom radu sve više oslanja na pomoć računala. Tijekom rješavanja konstrukcijskog zadatka konstruktor koristi razne programske aplikacije. Sva komunikacija tj. razmijena informacija između računala (programske aplikacije) i korisnika (konstruktora) ostvaruje se posredstvom korisničkog sučelja. Korisničko sučelje je korisnikov prozor u programsku aplikaciju i kao takovo ima najveći upliv na stvaranje korisnikovog "mišljenja" o programskoj aplikaciji.

Iz dana u dan svjedoci smo pojave sve bržih i jačih računala i sve "složenijih" programskih aplikacija koje, u dijelovima, pokrivaju gotovo čitavo područje rada konstruktora od koncipiranja ideje, preko proračuna do izrade tehničke dokumentacije. Moglo bi se zaključiti da je time posao konstruktorima olakšan, ali to nije slučaj. Jednim dijelom i zbog toga što je cjelovit i efikasan CAE (Computer Aided Engineering) sustav izuzetno kompleksan čak i na teorijskoj razini [1][2], a implementacija u praksi otvara još niz praktičnih problema. Koncipiranje strukture i problemi realizacije takovog sustava, koji će integrirati CAD (Computer Aided Design) i CAM (Computer Aided Manufacturing) sustave tema je velikog broja stručnih i znanstvenih radova [1][2][3][4][5] i istraživanja u području primjene računala u procesu razvoja proizvoda. Jedna od svrha svih tih istraživanja je daljnje povećanje produktivnosti u svim aktivnostima koje prate proces konstruiranja.

U većina “klasičnih” CAD programskih sustava na tržištu, metode umjetne inteligencije su implementiranu u vrlo malom obimu što je i najveći nedostatak tih sustava, a što je posljedica [6]:

- nemogućnosti obrade simboličkih informacija,
- nemogućnost reprezentacije znanja o konstruiranju,
- nepostojanje mehanizama zaključivanja,
- nepoznavanje formalnih zakona procesa konstruiranja, odnosno relacija između zahtijeva i konstrukcijskog rješenja.

Programske aplikacije koje su konstruktoru na raspolaganju pružaju različite mogućnosti za različitu cijenu. Većina komercijalnih aplikacija na tržištu je opće namjene tako da specijalizirana podru’ja npr. izrada CNC programa, analiza konstrukcije, optimalizacija konstrukcije nisu detaljno razrađena, u tom slučaju moraju se koristiti specijalizirane programske aplikacije. U većini specijaliziranih programskih aplikacija nedostaje neka od komponenti aplikacija široke namjene (nedostaje dio za kvalitetno modeliranje, postprocesor, predprocesor, mehanizmi za opisivanje crteža, itd.). U slučaju da nedostaje neka od potrebnih komponenti u programskoj aplikaciji konstruktor mora ili sam kreirati komponentu koja nedostaje ili mora koristiti komercijalne programske aplikacije. Komercijalne programske aplikacije ne podržavaju u cjelini proces konstruiranja već samo pojedine segmente procesa [6], te čine neovisne pasivne alate koji imaju ograničene mogućnosti međusobnog komuniciranja i prijenosa informacija.

Ukoliko bi se i mogao napraviti programski sustav koji bi podržavao u cjelini proces konstruiranja, jedna osoba ne bi bila u stanju jednako kvalitetno i produktivno koristiti se svim elementima sustava. Neki od problema koji bi se mogli pojaviti prilikom realizacije programskog sustava za podršku procesu konstruiranja su problem: komuniciranja između elemenata sustava, kontrole tijeka informacija, te problem pretvaranja podataka i njihovog prikaza. Problem koordinacije i interakcije s korisnikom potpuno je otvoreno pitanje koje se od sustava do sustava rješava na manje ili više (ne)sretan način.

Jedno od mogućih pristupa rješenju problema realizacije programskog sustava za podršku procesa konstruiranja je realizacija skupa alata koji "znaju" na koji način komunicirati s različitim programskim aplikacijama. Na taj način nema potrebe za izradom svih, nego samo pojedinih, modula potrebnih za provođenje procesa konstruiranja, a mogu se koristiti i programske aplikacije različitih programerskih kuća. No, problem je u kontroli i vođenju procesa konstruiranja. Problem kontrole i vođenja procesa konstruiranja prema [7] može se riješiti pomoću baze znanja, u kojoj su zapisani skupovi pravila i postupaka koji vrijede za određenu proizvodnu okolinu, odnosno vrstu proizvoda.

Mogućnost unapređenja CAD sustava temelji se na ideji ugradnje metoda umjetne inteligencije čime bi se sustav približio prirodnom tijeku promišljanja i obrade informacija. To znači da se koncipiranje modela prikaza tijeka konstruiranja prilagodi potrebi obrade numeričkih i simboličkih informacija na različitim razinama apstrakcije.

Jedan od važnijih dijelova takovog sustava je korisničko sučelje. Suprotno "opće prihvaćenom mišljenju" da korisničko sučelje čine samo sustav prozora, dugmad, i izbornika, korisničko sučelje obuhvaća i metode za:

- kontrolu ulaznih i izlaznih podataka (npr. područje vrijednosti, format zapisa),
- pretvorbu podataka (npr. pretvaranje geometrijskih podataka iz DXF formata zapisa u IGES),
- realilzaciju objašnjenja i pomoći (npr. objašnjenje rada dijelova programskog sustava ili pomoć pri odabiru neke opcije)

Znanost o konstruiranju je relativno mlada disciplina [1], te ne postoje zajednički usaglašeni stavovi među istraživačima o prikladnim metodologijama i prioritetnim smjerovima istraživanja. Takovo stanje uzrokuje određenu kaotičnost što otežava opisivanje i prepoznavanje podataka koji se generiraju u različitim fazama procesa konstruiranja. Također se može uočiti da nedostaje jedinstvena teorija CAD-a [8][1][2], te ne postoji potpuna reprezentacija strojarskih proizvoda, niti usaglašenost modela procesa konstruiranja strojarskih proizvoda.

U tijeku procesa konstruiranja generira se velika količina informacija. Budući da su ljudske mogućnosti percepcije ograničene mora se voditi računa o količini, obliku i mjestu prikaza pojedinih informacija. Prikaz velike količine nepovezanih podataka na zaslonu računala može zbuniti korisnika. Loše odabran oblik prikaza podataka može dovesti do previda važnih informacija. Prikaz naoko sličnih, a zapravo oprečnih informacija zajedno na zaslonu računala može navesti korisnika na pogrešan zaključak. U najboljem slučaju pojava neke od navedenih situacija može dovesti do pogreške u radu programskog sustava koja se tada dojavi korisniku te je on može ispraviti. U najgorem slučaju greška će se otkriti tek nakon završetka rada programskog sustava tijekom kontrole konstrukcijskog rješenja.

Upravo jedan od zadataka korisničkog sučelja je prikaz samo bitnih podataka i/ili podatke odabrane od strane konstruktora. Pri tome korisniku moraju biti dostupne sve informacije o tijeku procesa konstruiranja i trenutnom stanju programskog sustava.

Prema [7] i [9] jedno od mogućih rješenja problema informatizacije procesa konstruiranja je programski sustav koji objedinjuje komercijalne programske aplikacije kontrolirane i vođene bazom planova uz stalan nadzor korisnika. Programske aplikacije

se u većini slučajeva nalaze na lokalnom računalu. No to nije ograničenje, stupanj razvijenosti računalne tehnologije danas omogućava uporabu programskih aplikacija na drugim računalima putem mreže.

Grupni rad ili rad više korisnika, od kojih je svaki "specijalist" za određeni dio procesa konstruiranja, a koji obavlja svoj dio zadatka na dislociranom računalu, također je moguće ostvariti. Komunikacija između korisnika ostvaruje se preko lokalne ili globalne mreže ("Internet").

U radu je prikazano jedno od mogućih rješenja modela grafičkog korisničkog sučelja sustava za konstruiranje mehaničkih sklopova. Analizirani su i određeni zadaci i uloga korisničkog sučelja u programskom sustavu za podršku procesu konstruiranja te je prikazana i opisana funkcionalna struktura i dijelovi modela. Na osnovu opisa modela sučelja opisana je jedna verzija prototipa grafičkog korisničkog sučelja.

3.1. Opis zadatka

U ovom radu će se istražiti koncipiranje i kreiranje modela korisničkog sučelja primjenjivog za konstruiranje mehaničkih sklopova. Promatrajući proces konstruiranja kao proces generiranja informacija o proizvodu, može se zaključiti da je moguće ostvariti komunikaciju između različitih programskih alata i konstruktora s različitim oblicima prikaza podataka i tijeka procesa konstruiranja.

Model korisničkog sučelja dio je programskog sustava koje integrira skup raspoloživih programskih alata u konzistentno okruženje, temeljem planova konstruiranja zapisanih po određenoj sintaksi. Model programskog sustava, planovi konstruiranja i pojedni dijelovi programskog sustava, opisani su u sljedećim radovima [9] [7] [10] [11] [12] [3] [13] [14] [15] [16] [17].

Ovaj rad dio je ukupnog istraživanja na projektu razvoja modela ICAE (Intelligent Computer Aided Engineering) sustava. Značajke ovog sustava se trebaju približiti ideji "inteligentnog" CAE sustava. Model cjelokupnog sustava gradi se na modularnom principu. Djelove sustava čine raspoložive komercijalne kao i vlastite programske aplikacije. Korisničko sučelje povezuje dijelove programskog sustava u kompaktnu cjelinu. Tema ovog rada orijentirana je na koncipiranje korisničkog sučelja kao jednog od dijelova modela ICAE sustava [9]. Korisničko sučelje mora ispuniti sljedeće zadatke:

- ostvarivanje komunikacije s korisnikom,
- ostvarivanje komunikacije između dijelova programskog sustava,
- kontrola tijeka procesa konstruiranja,
- upravljanje radom programskog sustava,

- kontrola i prikaz informacija o odvijanju procesa konstruiranja,
- kontrola i prikaz informacija o radu programskog sustava,
- osiguranje interakcije s klasičnim programskim sustavima,
- transformacija zapisa i prijenos informacija.

Oblik prikaza podataka koje generiraju programske aplikacije (alati) u pojedinim fazama procesa konstruiranja je različit i slijedi iz razmatranja faza procesa konstruiranja. Oblik prikaza podataka koji su dobiveni kao rezultat rada dijelova programskog sustava koji nisu dio procesa konstruiranja slijedi iz razmatranja dijelova programskog sustava i informacija koje se generiraju u tijeku rada programskog sustava.

Ispravno koncipirano i realizirano korisničko sučelje omogućiti će korisniku da postane aktivni sudionik procesa konstruiranja, te da ima nadzor ne samo nad tijekom procesa konstruiranja i rada programskog sustava, već i nad svim dostupnim uređajima računala.

Kako dijelovi programskog sustava povezani preko korisničkog sučelja moraju djelovati kao cjelina, u tijeku koncipiranja svakog pojedinog modula sustava treba voditi računa o vezama s ostalim modulima, načinu prijenosa podataka i temeljima integracije sustava kao cjeline.

Kao rezultat uvodnih razmatranja može se zaključiti da okruženje za računalsku podršku konstruiranju treba integrirati vrlo raznorodne programske aplikacije: komercijalne grafičke pakete, baze podataka, ekspertne sustave i druge dostupne aplikacije. Takva integrirana okolina otvorena je za nadogradnju složenijim programskim alatima, u sljedećim fazama istraživanja, čime bi se sustav više približio ideji “inteligentnog”.

Kako je već navedeno, moguće je ostvariti komunikaciju između različitih programskih aplikacija i korisnika (konstruktora) uz različite oblike prikaza podataka i tijeka procesa konstruiranja. Iz toga proizlazi struktura modela korisničkog sučelja, čija bi funkcija bila ostvarenje komunikacije između različitih programskih aplikacija tj. njihova integracija u cjelinu te različiti oblici prikaza podataka i to onih vezanih za sam proces konstruiranja i onih vezanih za rad sustava.

Da bi se pristupilo koncipiranju i razradi modela korisničkog sučelja potrebno je analizirati postojeće modele i metode razvoja korisničkih sučelja.

U okviru koncipiranja i realizacije modela korisničkog sučelja treba odrediti:

- funkcionalne elemenate korisničkog sučelja; definiranjem okvira i ograničenja koncepcije modela koji se razvija obzirom na današnji stupanj informatičke tehnologije putem:
  - pregleda i analize mogućnosti i načina rada postojećih korisničkih sučelja,
  - pregleda postojećih standarda koji se koriste u izradi korisničkih sučelja,
  - razmatranja postojećih alata (razvojnih okolina) te njihovih značajki u izradi modela,
  - analize stanja u razvoju metoda umjetne inteligencije u području korisničkih sučelja,
  - razmatranja načina dobave i prikaza informacija na relaciji korisnik-računalo i računalo-korisnik,
  - analize mogućnosti postojećih računala i postojeće programske podrške za realizaciju predloženog modela,
- strukturu korisničkog sučelja temeljem zahtijeva u okviru mogućnosti proizašlih iz predhodne analize te realizirati:
  - razradu svih elementa i algoritama potrebnih za realizaciju modela,
  - model korisničkog sučelja u odabranom okruženju sa svim potrebnim procedurama i modulima,

U okviru razrade modela korisničkog sučelja treba:

- razmotriti primjenljivosti modela korisničkog sučelja u smislu ubrzanja i olakšanja procesa rješavanja zadanog problema, za razne klase zadataka u procesu konstruiranja,
- analizirati nedostatake i dati smjernice za unapređenje komunikacije između konstruktora i računala,
- predložiti metode umjetne inteligencije za unapređenje modela.

Budući da ne postoje standardizirana mjerila za ocjenu kvalitete korisničkih sučelja, razvijeni prototip modela korisničkog sučelja za programski sustav za podršku procesu konstruiranja će se vrednovati na osnovu subjektivne ocjene korisnika. Korisnici će tijekom testiranja korisničkog sučelja ocijeniti jednostavnost i lakoću uporabe, te prilagodljivost i funkcionalnost prototipa.

4. Proces konstruiranja

Sposobnost generiranja ideja dolazi s iskustvom, ali je potreban i talent. Za prosuđivanje ideja koristi se iskustveno znanje i znanje stečeno obrazovanjem. Znanje potrebno za generiranje i prosuđivanje (razmatranje) ideja ovisno je o domeni konstrukcijskog zadatka, dok je znanje o strukturiranju procesa konstruiranja većinom neovisno o domeni zadatka.

Svaki konstrukcijski zadatak može se riješiti na mnogo različitih načina, odnosno može rezultirati sa različitim strojnim sustavima ili sklopovima. Ta karakteristična mnogostrukost mogućih rješenja uvjetovana je količinom svojstava proizvoda koje se trebaju odrediti u postupku konstruiranja.

Svaki proces konstruiranja može se razložiti u manje cjeline (faze, dijelove procesa, etape, operacije) koje čine strukturu procesa.

Kompleksnost često međusobno proturječnih zahtjeva, rezultira najčešće višestrukim ponavljanjem određenih faza konstruiranja nakon početnog apstrahiranja i postavljanja pretpostavki. Iterativnost je stoga jedna od značajki konstruiranja.

Određeni misaoni postupci (intuicija, nastajanje ideje) koji se ne mogu racionalno objasniti imaju obilježja umjetničke kreativnosti. Ti postupci ne mogu se formalizirati u svrhu stvaranja cjelovitog teoretskog prikaza konstrukcijskog procesa.

Sa stanovišta teorije proizvoda mogu se navesti značajke koje utiču na procesa konstruiranja prema [18].

- nagli porast potreba uvjetovan tržišnim zakonitostima,
- period razvoja proizvoda je sve kraći,
- vijek trajanja proizvoda je sve kraći,
- javlja se pojam kritične brzine konstruiranja ,
- raste količina proizvoda u serijskoj i masovnoj proizvodnji,
- zahtjevi za kvalitetom također rastu,
- troškovi se moraju svoditi na minimum,
- najveći utjecaj na strukturu troškova ima konstrukcijsko rješenje proizvoda,
- produktivnost tehnologije raste daleko brže od produktivnosti onstruiranja.

Iz navedenih značajki može se uočiti mnogostrukost upliva na proces konstruiranja, kompleksnost procesa, a i širina potrebnih znanja. Detaljnija razmatranja upliva okoliša na proizvodni sustav s implikacijama na CIM, odnosno na proces konstruiranja i CAD mogu se naći u [3].

4.1. Metodičko konstruiranje

Prema [19] metodičkim se konstruiranjem nastoji primjenom znanstvenih metoda, razviti proces konstruiranja kao postupak koji omogućuje da se problematika konstruiranja rješava općenito, a ne kao problematika konstruiranja sasvim određenog stroja ili uređaja. Riječ je, zapravo o tome da se konstruiranje shvati kao proces u kojem se jednakim postupcima mogu rješavati različiti zadaci. U ovom poglavlju dan je skraćeni pregled modela procesa konstruiranja prema [1] kao jedna od smjernica koncipiranja sustava za računalnu podršku konstruiranju.

4.1.1. Deskriptivni, preskriptivni i računalni modeli procesa konstruiranja

Modeli procesa konstruiranja mogu se sa stanovišta pristupa podijeliti na deskriptivne, preskriptivne i računalne.

Deskriptivni modeli opisuju proces konstruiranja. Osnova razvoja deskriptivnih modela je proučavanje procesa kojima ljudi kreiraju konstrukcije, te koje strategije i metode koriste pri rješavanju zadataka. Razvoj deskriptivnih modela može se razvrstati na protokolarne studije, kognitivne modele i studije slučajeva (“case studies”).

Cilj je protokolarnih studija sustavno prikupljnje podataka o tome kako rade konstruktori, kao pojedinci ili u grupama. Problem je takvih studija što samo “snimanje” na određeni način i ometa konstruktora u radu. Isto tako teško je verbalno izraziti

geometrijsko promišljanje, a podatke o podsvjesnom promišljanju gotovo je nemoguće prikupiti.

Kognitivni modeli opisuju procese koji su osnova skupa ponašanja koje tvori vještinu. Takav se model određuje skupom mehanizama s definiranom funkcionalnošću i interakcijama. Najčešći je cilj istraživanja kognitivnih modela razvoj analognih računalnih modela procesa konstruiranja, tj. kognitivnih sustava. Kognitivni sustavi opisuju, odnosno emuliraju vještinu koji koriste ljudi pri rješavanju zadataka. Vještina ljudi se u takovim sustavima opisuje na razini funkcionalnih mehanizama, s mogućnošću objašnjavanja i predviđanja ponašanja pri rješavanju zadataka.

Studije slučajeva analiziraju proces konstruiranja u realnim okolnostima, na odabranim primjerima koji se razmatraju cjelovito, od zadatka do realiziranog proizvoda. Takve studije proučavaju osim tehničkih i sociološke aspekte konstruiranja. Rezultati takovih istraživanja pokazuju da konstruktori često ne razmatraju više koncepata potencijalnih rješenja zbog različitih razloga, npr. osobne sklonosti određenom rješenju, nedostatka vremena ili čak nisu ni svijesni drugih mogućih rješenja. U radu [20] se navodi da konstruktori ne posežu za alternativama i inovacijama, osim ako korišteno rješenje ne zadovoljava zahtjeve koji se ne mogu ispuniti nikakovim prilagođavanjem.

Preskriptivni modeli procesa konstruiranja mogu se podijeliti u dvije skupine: na modele koji opisuju kako treba provoditi proces konstruiranja, te na modele koji opisuju atribute koje konstrukcijska tvorevina treba posjedovati.

U prvu se skupinu mogu svrstati kanonski modeli i morfološki modeli. Kanonski se modeli primjenjuju najčešće u obrazovanju konstruktora.

Metode morfološke analize često se koriste u preskriptivnim modelima, a osnivaju se na slijedećim pretpostavkama:

- svaki se složeni konstrukcijski model može podijeliti na konačan broj podzadataka,
- svaki se podzadatak može razmatrati izolirano, uz privremeno zanemarivanje međudjelovanja s ostalim podzadacima,
- svi se podzadaci i pripadna rješenja mogu prikazati u morfološkoj tablici,
- cjelokupno rješenje zadatka može se naći kombinacijom rješenja pojedinih podzadataka,
- način kombiniranja rješenja podzadataka u cjelokupno rješenje nije ograničen.

Drugu skupinu preskriptivnih modela procesa konstruiranja čine modeli koji opisuju atribute konstrukcije, te time propisuju svojstva koja treba posjedovati

konstruirana tvorevina, a ne postupak kojim se tvorevina određuje. Ovi se modeli baziraju na aksiomatskim teorijama i svaki je model samostalna cjelina [9].

Računalni modeli procesa konstruiranja podrazumijevaju programima implementirane metode kojima računalo može riještiti određene zadatke. Računalni modeli mogu dijelom biti razvijeni na temelju promatranja ljudskog načina promišljanja zadatka, ali takova sprega nije uvijek nužna. Uspješni računalni modeli mogu ponekad i sugerirati postupke "ljudskog" načina izvođenja procesa konstruiranja.

U području računalnih modela treba razlikovati metode koje služe u projektiranju (prvenstveno s ciljem odlučivanja), metode koje analiziraju, tj. daju informacije koje su podloga za donošenje odluka, te "klasične" CAD sustave.

Sustavi za projektiranje baziraju se na ideji da se funkcionalni zahtjevi koji se postavljaju na proizvod i forma proizvoda mogu povezati. To vrijedi za dobro definirane zadatke s uskim područjima primjene. Optimalizacijski sustavi su jedan od primjera metoda koje se u općim slučajevima mogu primjeniti za odlučivanje.

"Klasični" CAD sustavi mogu se obzirom na namjenu i stupanj postignute automatizacije podijeliti na sustave opće namjene i sustave ograničenog područja primjene (za razvoj određene vrste proizvoda).

Računalni modeli mogu se razvrstati prema primjeni za slijedeće klase konstrukcijskih problema (i/ili zadataka):

- parametarsko konstruiranje,
- određivanje konfiguracije sklopova,
- koncipiranje na temelju funkcionalnih zahtjeva.
- rješavanje distribuiranih zadataka.

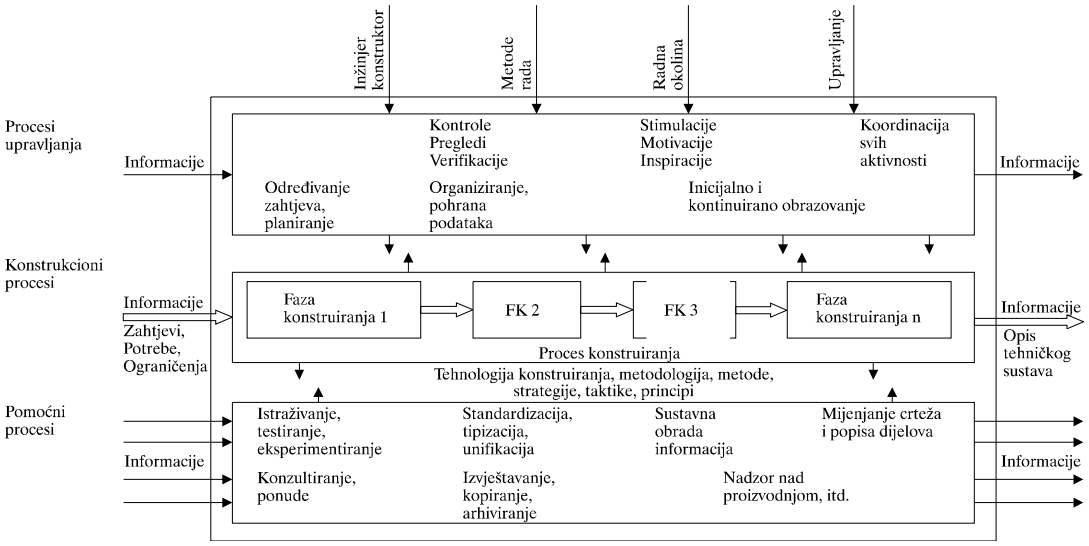
4.2. Opći model procesa konstruiranja

Zakonitosti koje vladaju u procesu konstruiranja kao kreativnoj ljudskoj djelatnosti, nastoje se opisati teorijama čija su polazišta uvjetovana okolinom i svim njenim uplivima.

Opći model konstruiranja koji je postavio Hubka 1973. godine daje pregled skupa aktivnosti, vrsta procesa i upliva na proces konstruiranja. Razvojem općeg modela konstruiranja autor je tijekom vremena postavio teoriju tehničkih sustava, te unutar nje noviji "opći proceduralni model konstruiranja". Navedeni modeli procesa konstruiranja tipični su primjeri kanonskih preskriptivnih modela.

Opći model procesa konstruiranja (slika 2-1) može se interpretirati na slijedeći način:

- konstruiranje je proces transformiranja informacija od zahtjeva kupaca do potpunog opisa predloženog tehničkog sustava,
- prikazuje se osnovna struktura procesa, uključujući regulacijske, kontrolne i pomoćne procese,
- kao direktni operatori, konstruktori i njihova sredstva za rad izvode akcije (efekte) na skupu informacija (operanada konstrukcijskog procesa),
- prikazuje se utjecaj ostalih različitih faktora na proces konstruiranja (metode rada, radna okolina, upravljanje).



Slika 4-1 Opći model procesa konstruiranja

4.3. Konstruiranje kao rješavanje zadatka

Prema [21][22] zadatak procesa konstruiranja je pretvorba postavljenih zahtjeva u opis željenog strojnog sustava. Drugim riječima potrebno je definirati i opisati strojni sustav (proizvod) koji će proizvoditi željene učinke i pri tome zadovoljavati zahtijevana svojstva. Osim postavljenih zahtjeva potrebno je voditi računa i o ostalim aspektima tehničke i ekonomske naravi - npr. tehnologičnost, troškovi, itd. - što vrijedi i za sam proizvod, i za njegovo funkcioniranje u životnom vijeku.

Nije na odmet razmotriti moguću dvosmislenost, odnosno poistovjećivanje pojmova problema i zadatka. Pojam rješavanja problema trebalo bi vezati za situacije u kojima na

početku nije poznata metodologija rješavanja. U procesu rješavanja zadatka metodologija je poznata, odnosno korištenjem poznatih metoda napreduje se od polazišnog stanja k cilju - rješenju. Riješiti problem, znači da istraživanjem (najčešće heuristikom, odnosno intuicijom) treba prvo definirati (pretpostaviti, pa isprobati) metodologiju, jer se način rješavanja (a često i sam uzrok problema) ne može odmah nazrijeti. U literaturi većinom nema jasne distinkcije između procesa rješavanja zadatka i rješavanja problema - pretežno se to svodi na razvrstavanje konstrukcijskog procesa obzirom na količinu poznatih polazišnih komponenti i njihovih relacija u mehaničkom sustavu kojeg treba definirati.

Također bi trebalo napomenuti da se u praksi često javljaju situacije da zadatak, odnosno lista zahtjeva na proizvod nisu točno ili potpuno definirani što može rezultirati nesporazumima i posljedično povećanim troškovima i dugim rokovima. Za ilustraciju dana je struktura definicije zadatka prema [18].

- definiranje ciljeva - ima prije svega strateški smisao, a daljnja razrada ima taktički smisao,
- opis problema na slobodniji način,
- navesti što su nužni zahtjevi, a što željeni,
- definiranje uvjeta za realizaciju zadatka uz provjeru da li su predviđeni uvjeti točni,
- razmatranje mogućnosti daljnjeg razvoja rješenja (da li se zadatak izvršava kao nešto konačno ili u prijelaznom obliku),
- definirati koja svojstva rješenja mora imati,
- definirati koja svojstva rješenje ne smije imati,
- rašćišćavanje zadatka - provjera da li je sve jasno i potpuno, da li je nešto inkompatibilno.

Ako se zadatak definira u fazi nuđenja proizvoda, najčešće treba još uključiti i predprojekt i kalkulaciju.

Prema polazištu zahtjeva koji definiraju zadatak, zadaci se mogu razlučiti na:

- "interne" razvojne zadatke koji su potaknuti unutar samog proizvodnog okruženja - najčešće temeljem povratnih sprega u informacijskim tokovima,
- zadatke potaknute indirektnim zahtjevima tržišta - temeljem praćenja stanja tržišta i konkurencije,
- zadatke potaknute direktnim zahtjevima tržišta odnosno narudžbom.

4.3.1. Faze rješavanja konstrukcijskog zadatka

Zadatak koji se prenosi konstruktoru, je pored ostalog, rezultat izbora koji tim planera (rukovodioci, prodaja, razvoj, proizvodnja) vrši prema raznim kriterijima (tržište, želje kupaca, troškovi, kapaciteti, rokovi, rizik, itd). Prema [19] proces konstruiranja može se podijeliti na faze od kojih je prva defniranje zadatka (preko liste zahtijeva):

Koncipiranje predstavlja onaj dio procesa konstruiranja pri kojem se nakon raščišćavanja svih zahtjeva vezanih za zadatak utvrđuje principjelno rješenje zadatka, traženjem odgovarajućih metoda i principa djelovanja. Dobivena rješenja vrednuju se prema kriterijima danim u listi zahtjeva.

Projektiranje je faza procesa konstruiranja u kojoj se utvrđuje funkcionalno, strukturalno i ekonomsko rješenje zadatka u takvom opsegu da je moguća daljnja konstruktivna razrada. Izrađenost projekta treba biti takva da se može izvršiti kalkulacija i nuđenje. Projekt treba obuhvatiti sve što sustav sadrži, ali ne daje odgovor kako to treba izraditi. Pošto se iz projektnog rješenja odstrane slaba mjesta, pristupa se tehničkom i ekonomskom vrednovanju i nakon toga optimiranju projektnih detalja.

Konstrukcijska razrada ima zadatak definirati informacije kako treba konstrukciju izraditi odnosno kako treba fizički izgledati kao gotov proizvod. U toj fazi detaljno se razrađuje sva potrebna tehnička i tehnološka dokumentacija za odabranu varijantu rješenja.

U procesu rješavanja zadatka konstruktor ne postavlja uvijek jasne granice između navedenih faza, a isto tako često može i naizmjence obavljati aktivnosti iz različitih faza. Određene vrste operacija odnosno aktivnosti mogu se na isti način odvijati u svim fazama. Iz tih razloga sustav za podršku planiranju konstrukcijskog procesa treba pokušati modelirati na taj način da se može primjeniti u svim fazama procesa konstruiranja, odnosno sinatksa i elementi zapisa plana ne trebaju ovisiti o fazi procesa konstruiranja u kojoj se primjenjuju.

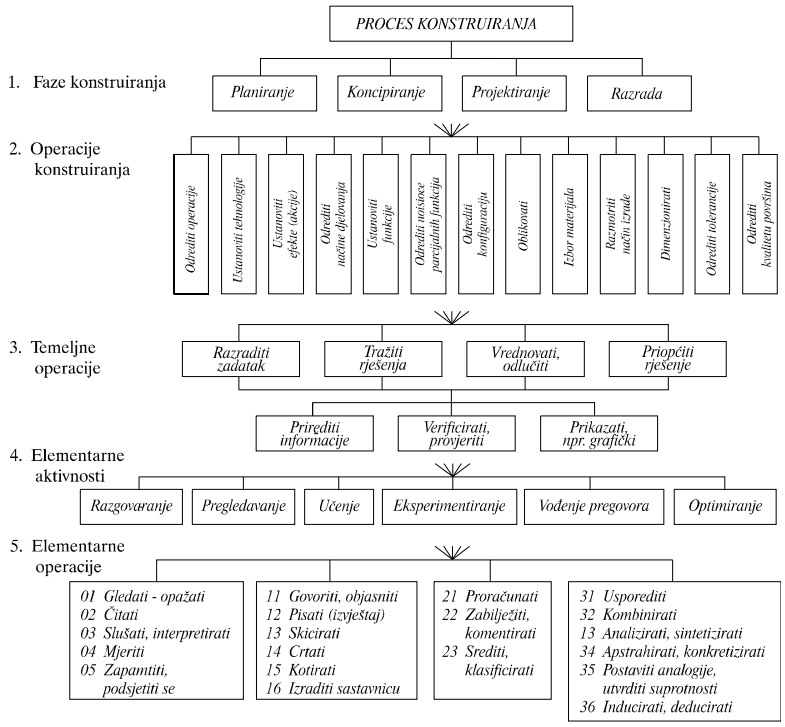
4.3.2. Struktura operacija u procesu konstruiranja

Svaki konstrukcijski proces može se raščlaniti na nekoliko kompleksnih faza, odnosno na operacije i korake. Struktura konstrukcijskog procesa može se prikazati kao niz strukturnih elemenata koji leže na različitim hijerarhijskim razinama. Pravila strukturiranja mogu poslužiti kao polazišta za upravljanje i vođenje, kao i planiranje konstrukcijskog procesa. U prikazu (Slika 4-2) strukture procesa konstruiranja prema modelu opisanom u točki 4.2 raščlanjene su operacije i aktivnosti procesa [21].

Prikaz strukture konstrukcijskog procesa može se promatrati na dva načina:

- kao hijerarhija kompleksnosti u aktivnostima konstrukcijskog procesa - svi elementi (operacije) iz nižih razina sadržani su u svakom od elemenata u višim razinama,
- kao grupe aktivnosti koje se ciklički ponavljaju do postizanja ciljeva.

Razmatranjem prikaza strukture operacija nameće se ideja o razvoju skupa programskih alata za podršku obavljanju onih operacija (između navedenih) koje se mogu algoritmizirati, a nisu (kao cjelina) podržane današnjim CAD sustavima. Naravno, takovi programski alati trebali bi omogućiti obavljanje operacija na poopćenoj razina apstrakcije, tj. neovisno o vrsti proizvoda i konstrukcijskog zadatka, uz mogućnost prilagodbe uvjetima okruženja u kojem se koriste. Razvoju takvih vrlo složenih programskih alata treba prethoditi razvoj integrirane programske okoline za modeliranje planiranja, praćenja i kontrole tijeka odvijanja akcija, odnosno skupa procedura koje obavljaju operacije.



Slika 4-2 Struktura aktivnosti u konstrukcijskom procesu prema [21]

Modeliranje poopćenih operacija može se realizirati uzastopnim pozivanjem niza programskih aplikacija. Uporabom postojećih programskih procedura ili razvojem novih, uz osiguranje integralnog korisničkog i programskog sučelja za prijenos podataka između modula, može se modelirati računalska podrška izvođenju određene konkretne operacije,

koja vrijedi za određeni zadatak ili klasu zadatka, odnosno konstrukciju, i to u uvjetima okruženja u kojem se odvija konkretan konstrukcijski proces.

4.3.3. Ograničenja i odluke u procesu konstruiranja

Proces napredovanja od početne specifikacije (definicije zadatka ili problema) do rješenja, odnosno kompletnog skupa informacija o željenom proizvodu može se promatrati kao niz koraka u kojima se izmjenjuju procesi obrade informacija i donošenja odluka. Svaki od koraka može se naznačiti nekom odlukom koja na bilo koji način mijenja stanje unutar skupa informacija o proizvodu. Skup informacija o proizvodu čine svi generirani crteži, modeli, analize, bilješke i prikupljeno znanje tokom procesa konstruiranja.

Prema [23] mogu se razlučiti dva različita gledišta o tome kako proces konstruiranja napreduje po koracima, odnosno od jednog stanja do slijedećeg.

Po jednom pristupu, opis proizvoda (konstrukcija) evoluira tokom kontinuiranog (cikličkog) procesa usporedbe između trenutnog stanja definiranosti skupa informacija o proizvodu i cilja, tj. skupa zahtjeva na konstrukciju definiranih konstrukcijskim zadatkom. Takav pristup implicira točno poznavanje svih zahtjeva na početku rješavanja zadatka kako bi se jasno mogla definirati razlika u odnosu na trenutno stanje definiranosti konstrukcije. Razlika tih dvaju stanja tada kontrolira proces konstruiranja. Međutim, za većinu konstrukcijskih zadataka i/ili problema, takav pristup je previše jednostavan, jer u početku procesa nisu svi zahtjevi precizno definirani, odnosno mogu se u toku procesa modificirati, pa ciljno stanje konstrukcije ne može na početku biti potpuno poznato.

Po drugom pristupu, na početku rješavanja zahtjevi na konstrukciju ograničavaju skup mogućih rješenja na podskup svih mogućih rješenja. Kako proces konstruiranja napreduje, nova ograničenja se dodaju da bi dalje reducirala moguća rješenja, koja se kontinuirano eliminiraju do jednog konačnog rješenja. Drugim riječima, konstruiranje je sukcesivno razvijanje i primjena ograničenja dok ne preostane samo jedno rješenje.

Ograničenja koja se dodaju u tijeku procesa konstruiranja proizlaze iz: općeg stručnog znanje konstruktora kao, znanja iz domene rješavanja zadatka te ograničenja koja se primjenjuju u tijeku konstrukcijskog procesa. Kako svaki konstruktor raspolaže sa djelomično različitim znanjem, primjenjivati će i različita ograničenja, pa će svaki riješiti zadatak na svoj, jedinstveni način. Odluke definiraju ograničenja i na taj način mogu utjecati na slijedeće odluke u daljnjem tijeku procesa. Može se reći da se većina ograničenja temelji na rezultatima konstrukcijskih odluka. Zbog toga je jedna od esencijalnih osobina konstruktora sposobnost donošenja odluka, ali na temelju dobro razrađenih kriterija, i u trenutku kad raspolaže s dovoljnom količinom informacija.

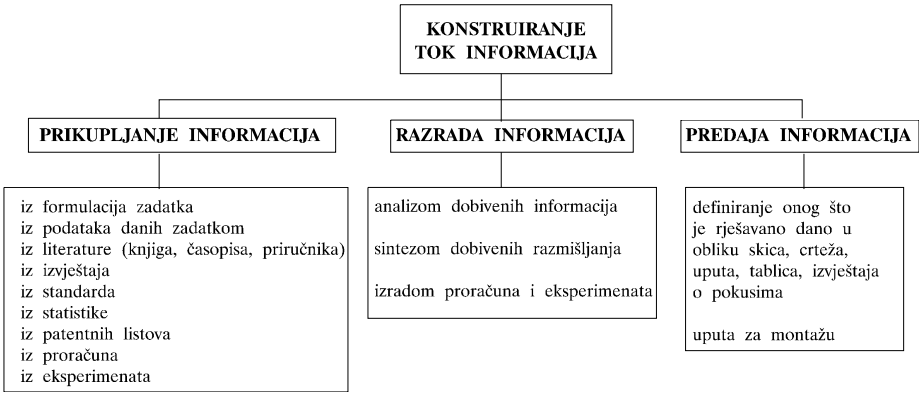
4.4. Informacija u procesu konstruiranja

Prikupljanje informacije je jedna od djelatnosti konstruktora. Značenje informacija je toliko da vrijeme utrošeno za njihovu dobavu, vrednovanje i klasifikaciju utječe na vrijeme potrebno za konstruiranje proizvoda. Prema [19] sve kraća vremena trajanja proizvoda uvjetuju skraćenje vremena potrebnog za izradu novog proizvoda što znači da se konstruktivni zadaci moraju s postojećim kapacitetima riješiti brže nego ranije. Ovi uticaji otežavaju prikupljanje informacija tako da je problem dobave, vrednovanja i klasifikacije informacija jedan od odlučujućih faktora u smanjenju vremena konstruiranja. Detaljna analiza informacija u tijeku procesa konstruiranja je opsežan zadatak koji prelazi granice ovog rada.

Posjedovanje ispravne informacije u pravo vrijeme je od vitalnog značaja gotovo u svim granama ljudskog djelovanja [24][22]. Poteškoće u dobavi i pripremi informacija javlja se u slučaju da je:

- Količina informacija i media za prijenos informacija obimna.
- Samo dio informacija koristan.
- Životni vijek nekih informacije kratak, tj. informacija postaje zastarjela u kratkom vremenskom periodu.
- Terminologija nedovoljno usaglašena.
- Vrijeme za proučavanje informacija premalo.
- Obrada i dobava informacija skupa.
- Znanje o informacijama i dokumenataciji nedostatno, što dovodi do loše uporabe.

Konstrukcijski proces možemo promatrati i kao tijek informacija (Slika 4-3) pri ćemu oblik i količina generiranih i potrebnih informacija u pojedinim fazama procesa konstruiranja ovise i o konstrukcijskom zadatku (nova konstrukcija, varijantna konstrukcija, ...).



Slika 4-3 Konstruiranje prikazano kao tijek informacija [19]

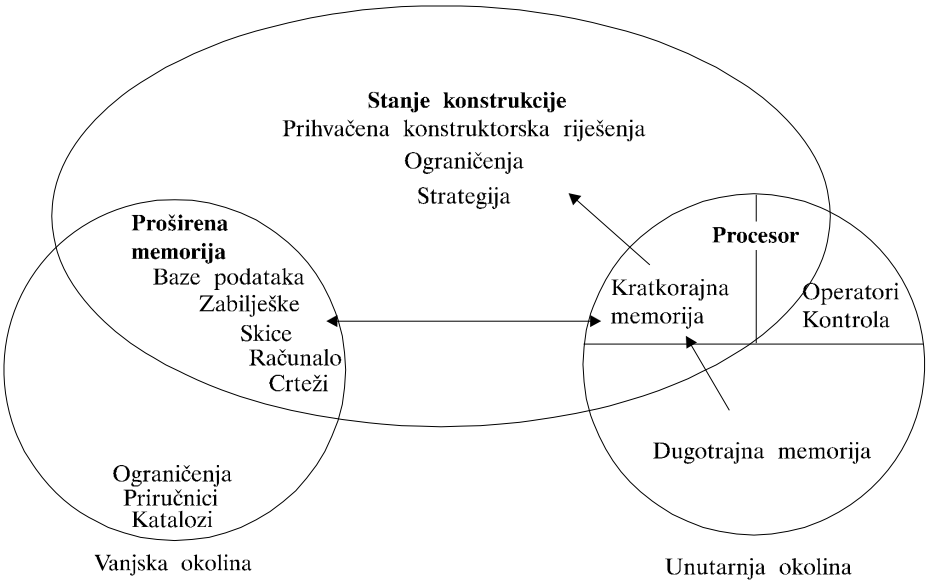
Pod tehničkim informacijama podrazumijevaju se sve zapisane informacije, iskustvo i grupno znanje neophodno za konstruiranje. Ove informacije sadrže sve od zakona prirode do podataka vezanih za određenu operaciju u pojedinom tehničkom sustavu. Uporabljivost informacije ovisi o sljedećim osobinama:

- Ispravnost sadržaja.
- Razumljivost.
- Jedinstvenost, jasnoća i točnost.
- Dostupnost u traženom vremenu.
- Mogućnost provjere.
- Potpunost.
- Oblik informacije.
- Način prijenosa.
- Izvor informacije.
- Jednostavnost pristupa informaciji.

Stanje informacije je presudno za mogućnost pristupa i dobave informacije. Možemo razlučiti dva stanja informacije:

- Varijabilna informacija (u ljudskoj memoriji), teško dostupna.
- Stalna informacija, zapisana u određenom obliku na nekom mediju.

Dijelovi stalne informacije su dostupni samo ukoliko su ispravno dokumentirani i katalogizirani. Na slici 2-4 prikazan je tijek i oblik informacija u tijeku procesa konstruiranja, dobivenih na osnovu ekperimenata opisanih u [25].



Slika 4-4 Tijek i oblik informacija u tijeku procesa konstruiranja.

4.4.1. Prikaz informacija

Tijekom procesa konstruiranja inžinjer konstruktor koristi različite fizikalne, matematičke ili konceptualne realizacije za prikaz stanja procesa konstruiranja na određeni način. Različiti prikazi konstrukcije uvjetuju prikaze informacija na različitim razinama apstrakcije. Prikazi različitih razina apstrakcije se prema [22] mogu podijeliti na:

- prikaz pomoću **ikona**, pri čemu ikone predstavljaju prikaz stvarnog stanja tj. pravog oblika informacija (skice, crteži, fotografije, modeli),
- **simbolički**, prikaz preko standardnih, opće usvojenih simbola (jezik, matematički zapisi, analogije),
- **dijagramski**, prikaz pomoću grafikona, dijagrama (za prikaz relacija), shema,
- prikaz **ponašanja**, ovaj prikaz uključuje djelom prototipove, rezultate modela u zračnim tunelima,
- **strukturni** prikaz, koristi se za kontrolu, izradu NC (Numeric Control) kodova, sklapanje ("assembly"),
- **makete**, koriste se za valorizaciju estetskih osobina modela.

Tijekom koncipiranja i razrade modela korisničkog sučelja ograničit ćemo se samo na prikaze podataka pomoću ikona i grafikona.

"Način i značenje prikaza informacija manje ili više realnih objekata u svrhu komuniciranja igra toliko značajnu ulogu u procesu konstruiranja da se može prikazati kao jedan od uticajnih faktora u procesu konstruiranja." (Hubka V).

Podaci se u najvećem broju slučajeva prikazuju u tekstualnom obliku. Ovakav prikaz je najčešće dostatan za uvid u stanje i vrijednost podataka. No, u određenom broju slučajeva tekstualni način nije dovoljan za prikaz potpune slike o podacima. U tom slučaju se koristi grafički način prikaza informacija.

Za generiranje grafičkih prikaza potrebno je znatno više podataka nego za tekstualni prikaz. Pojedine informacije se ne mogu prikazati grafički, zbog nedostatka podataka o njima (da li je podatak temperatura, tlak, promjer ...) ili zbog prirode informacije (širina ležaja). Informacije se mogu prikazati na sljedeće načine:

- tekstualno,
- grafički:
  - tablično,
  - dijagramski,
  - crtežom.

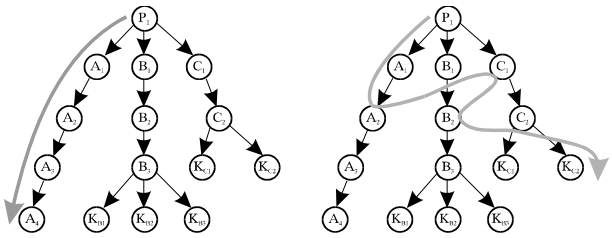
Način prikaza informacija ovisi o prirodi informacija i o zadanom, odnosno podrazumijvajučem obliku prikaza. Pod podrazumijevajučim oblikom prikaza se smatran unaprijed određen prikaz podatka, koji će biti korišten u slučaju da niti jedan drugi nije odabran.

5. Prikaz procesa konstruiranja planom

Izlaganje koncepcije strukture plana za prikaz procesa konstruiranja temelji se na rezultatima istraživanja opisanim u [9][7][10][11][26][15][27]. Koncepcija programskog alata za generiranje planova, odnosno struktura baze scenarija i sam postupak generiranja plana opisani su u [7].

5.1. Struktura plana

Kao model prikaza tijeka procesa konstruiranja odabrano je opće stablo [9] u kojem čvorovi predstavljaju operatore plana. Korištenjem hijerarhijske mreže može se realnije modelirati tijek procesa konstruiranja. Međutim i uz usvojeni način zapisa procesa postoji mogućnost da konstruktor tijekom eksploatacije plana aktivira čvorove plana po svom nahođenju. Osnovni način izvođenja plana je slijedni - po predviđenim putanjama, međutim posredstvom modula za kontrolu izvođenja, konstruktor može u svakom trenutku prekinuti takav proces i aktivirati neki od čvorova koji nisu na istoj putanji. Na taj se način plan zabilježen kao stablo može eksploatirati kao mreža (Slika 5-1).



Slika 5-1: Načini korištenja plana

Eksploatacija takvog plana podrazumijeva da se iz zadovoljenih početnih uvjeta (u korijenskom čvoru) izvodi plan koji će rezultirati stanjem zadovoljenog cilja. Pri tome taj cilj može biti samo jedan od podciljeva u ukupnoj strukturi konstrukcijskog zadatka. U drugom poglavlju već je napomenuto da nije uvijek nužno postavljati plan rješavanja na najvišeoj razini apstrakcije zadatka jer se zadatak može podiljeliti na hijerarhiju planova i podplanova, sa svojim parcijalnim ciljevima.

Izvođenjem, odnosno eksploatacijom plana upravlja poseban programski modul koji je također sastavni dio ekspertnog CAD sustava. Kontrola i upravljanje izvođenjem plana koncipirani su tako, da samo unaprijed predviđene situacije u planu određuju način izvođenja plana.

Svaki čvor u stablu plana jednoznačno je opisan svojim atributima. Pod pojmom atributi čvora podrazumijevaju se sve one oznake, veličine ili skupovi koji jednoznačno opisuju čvor u strukturi plana konstrukcijskog procesa. Za svaki pojedini čvor u strukturi plana potrebno je, u potpunosti, definirati sve njegove atribute

Definirani atributi čvorova plana su:

- akcija,
- ulazne informacije,
- izlazne informacije,
- relacije ograničenja kojima su podvrgnute izlazne informacije,
- relacije odlučivanja za odabir putanje nastavka procesa.

Pojam čvora u planu zapravo označava mjesto primjene operanda, odnosno izvođenja operacije koja se realizira pozivom određene akcijske funkcije.

Pojam akcijske funkcije je glavni element strukture plana - ona predstavlja operaciju prerade informacija na nekom podskupu skupa informacija o proizvodu koji se konstruira. Zbog toga su uz svaku akcijsku funkciju vezani skupovi ulaznih i izlaznih atributa koji pridruživanjem akcijske funkcije čvoru postaju atributi čvora. U trenutku izvođenja određenog čvora u planu, skupovi ulaznih i izlaznih atributa pripadne akcijske funkcije postaju podskupovi cjelokupnog skupa informacija o proizvodu koji se konstruira.

Proces konstruiranja, odnosno rješavanja zadatka, napreduje kroz postavljanje i provjeru ograničenja, te donošenje odluka. Strukturom plana predviđeno je za sada postavljanje ograničenja unaprijed (ne i u toku izvođenja plana), a isto tako unaprijed se određuju i pravila na temelju kojih se donose odluke. (Već je napomenuto da samo unaprijed predviđene situacije u planu određuju način izvođenja plana). Nakon izvođenja akcijske funkcije, slijedeća operacija unutar jednog čvora je provjera predviđenih

ograničenja na skupu izlaznih atributa. Promjena ili dodavanje ograničenja za sada nije predviđeno, s time da konstruktor može naložiti sustavu za izvođenje plana da određena ograničenja ignorira. Nakon provjere ograničenja, slijedeća operacija u čvoru je donošenje odluke koji će se od čvorova neposrednih sljedbenika dalje izvesti, odnosno bira se daljnja putanja u stablu plana. Detaljniji opis procesa izvođenja plana dan je u petom poglavlju.

5.2. Sintaksa plana

U ovom će se poglavlju izložiti sintaksa plana konstruiranja razvijena i definirana u sklopu projekta razvoja modela "inteligentnog" CAD sustava [9]. Temeljem predložene sintakse može se realizirati podsustav za upravljanje planom, a time indirektno i svi ostali podsustavi. Zapis cijelog plana [9][10][27] predstavlja niz slogova varijabilne dužine koji se zapisuju u obliku sekvencijalne datoteke. Temeljni elementi strukture plana opisani su ključnim riječima, koje služe za prepoznavanje.

Atributi čvora zapisani su u obliku slogova koji slijede iza odgovarajuće ključne riječi. Zapis svake vrste atributa sastoji se od predviđenih elemenata odvojenih odjelnim znakovima.

Osnovna sintaksa slogova s ključnom riječi je:

$KLJUČNA_RIJEČ: sadržaj

- gdje sadržaj predstavlja informacije koje ovise o ključnoj riječi.

Poziciju čvora unutar plana procesa konstruiranja određuje oznaka čvora, oznaka pripadnog prethodnika (nadređenog čvora) i oznake neposrednih sljedbenika.

Naziv izvršne (akcijske) funkcije pridružuje se oznaci čvora. Akcijska funkcija može biti bilo kakav programski alat koji na temelju skupa ulaznih informacija (preduvjeta) generira skup izlaznih informacija odnosno efekata.

Skupovi ulaznih i izlaznih atributa, te relacije odluka i ograničenja također se pridružuju čvoru. Navedeni skupovi atributa strukturirani su u obliku tablica, odnosno niza slogova koji sadrže predviđene elemente odvojene odjelnim simbolima. Time su određeni svi potrebni atributi čvora u planu konstruiranja.

Zapis plana u obliku sekvencijalne datoteke mora se prije samog izvođenja plana prevesti u binarni oblik koji će biti učitan u memoriju zajedno s programskim modulom koji kontrolira izvođenje plana. Iz tog razloga skupovi atributa čvorova strukturirani su u

obliku tablica. U prvoj fazi prevođenja plana rezervira se memorijski prostor za zapis čvorova i tablica njihovih atributa, a u drugoj fazi kreiraju se strukturne veze među čvorovima plana. Odjelni simboli, ključne riječi i redoslijed elemenata sintakse određeni su tako da omogućuju prevođenje plana sekvencijalnim čitanjem slogova datoteke.

Datoteka zapisa plana sadrži tri vrste slogova:

- slogove koji sadrže samo jednu od ključnih riječi,
- slogove koji sadrže kjučnu riječ i samo jedan atribut,
- slogove koji sadrže niz atributa odvojenih odjelnim simbolima.

U nastavku je iznesen opis strukture pojedinih skupova atributa čvora po redoslijedu kojem se za svaki čvor navode u zapisu plana.

5.2.1. Oznaka čvora

Oznaka čvora je simbolička oznaka - znakovni niz najveće dužine 16 znakova. Oznaka čvora dolazi iza ključne riječi $NODE_ID. Nedozvoljeni znakovi u oznaci čvora su "/" i prazno mjesto. U planu konstrukcijskog procesa svaka se oznaka čvora smije pojaviti samo jednom. Namjena svakog čvora u planu može se opisati komentarom koji slijedi iza oznake čvora.

5.2.2. Oznaka nadređenog čvora

Za svaki čvor potrebno je naznačiti oznaku nadređenog čvora kako bi se mogla formirati struktura stabla plana u toku prevođenja plana u binarni oblik. Oznaka nadređenog čvora slijedi iza ključne riječi $PARENT_ID. Struktura stabla formira se pomoću liste pokazivača. Za svaki čvor treba definirati pokazivač na nadređeni čvor i niz pokazivača na sve podređene čvorove. U slučaju korijenskog čvora iza kjučne riječi $PARENT_ID piše se oznaka samog korijenskog čvora, odnosno njemu se postavlja pokazivač na samog sebe.

5.2.3. Oznake podređenih čvorova

Oznake podređenih čvorova pišu se kao niz slogova koji slijedi iza ključne riječi $CHILDREN. Pri tome treba napomenuti da se ne navode svi podređeni čvorovi određenog čvora, nego samo neposredni potomci. Za krajnje čvorove u strukturi stabla ne piše se iz ključne riječi ništa, a njima se pokazivači definiraju kao NULL pokazivači (NULL pointers). Potrebno je naglasiti da se oznaka bilo kojeg čvora iz plana smije pojaviti najviše jednom u zapisu plana kao oznaka podređenog čvora. Kada ne bi bio

ispunjen taj uvjet, to bi značilo da plan više nema strukturu stabla, nego hijerarhijske mreže jer bi neki čvorovi imali više od jednog nadređenog čvora.

5.2.4. Akcijska funkcija

U planu konstrukcijskog procesa potrebno je, za svaki čvor, zapisati naziv funkcije koja će se izvoditi u tom čvoru. Naziv akcijske funkcije je maksimalne duljine 32 znaka, a navodi se iza ključne riječi $ACTION_FUNCTION. Naziv akcijske funkcije može, ali i ne mora biti isti kao ime datoteke koja sadrži izvršni programski kod akcijske funkcije. Iza naziva akcijske funkcije slijedi komentar sa opisom namjene akcijske funkcije.

5.2.5. Odluke

U tablice odlučivanja zapisuje se skup pravila na temelju kojih se vrši izbor slijedećeg čvora za izvršavanje, između sljedbenika trenutno aktivnog čvora. Tablica odlučivanja atribut je čvora koji može biti prazan skup u slučajevima kada određeni čvor ima samo jedan ili niti jedan podređeni čvor.

Iza ključne riječi $DECISION_TABLE. slijede uređeni nizovi slijedećeg oblika:

TV, AV, OP, DG, GG, OZN, OPIS

gdje su:

TV	tip varijable (I-integer), (D-double float), (C-character string)
AV	adresa varijable
OP	operator (<, <=, ==, >, >=, !=, range (interval))
DG	donja granica (konstanta ili adresa varijable)
GG	gornja granica (0 - za sve vrste operatora osim range)
OZN	oznaka čvora koji će se slijedeći izvesti ako je uvjet istinit
OPIS	komentar, opis pravila

Kao varijabla u tablici odlučivanja može se koristiti bilo koja od varijabli iz tablica izlaznih atributa nadređenih čvorova (po putanji do korijena) trenutno aktivnog čvora (kojem pripada tablica odlučivanja). Isto tako mogu se koristiti i varijable iz tablice ulaznih ili izlaznih atributa trenutno aktivnog čvora. Zbog toga se adresa varijable zapisuje u obliku **naziv_čvora / broj_sloga / X**. Broj sloga je redni broj varijable u pripadnoj tablici čvora, a **X** može biti “I” ili “O”, tj. označava da li se radi o tablici izlaznih ili ulaznih atributa.

5.2.6. Ulazni atributi

Skup ulaznih atributa sadrži podatke o varijablama čije vrijednosti će se predati akcijskoj funkciji u trenutku poziva. To u pravilu ne mora biti kompletan skup ulaznih podataka koji akcijska funkcija kao programska procedura zahtijeva, naime preostale ulazne podatke može osigurati u interakciji s korisnikom.

Članovi skupa ulaznih atributa slijede kao niz slogova iza kjučne riječi $INPUT_SET_TABLE. Skup ulaznih atributa sastoji se od slijedećih elemenata:

TV, AV, SI,OPIS

gdje su:

TV	tip varijable (I-integer), (D-double float), (C-character string)
AV	adresa varijable
SI	simbolička oznaka varijable
OPIS	pojam koji predstavlja varijabla

Sve varijable u tablicama ulaznih atributa svih čvorova osim korijenskog moraju biti adresirane. Adresa je zapravo pokazivač na određenu varijablu iz tablice izlaznih atributa nekog od nadređenih čvorova. Pri tome dolaze u obzir nadređeni čvorovi po putanji od onog kojem tablica pripada, do korijena. Adresa varijable zapisuje se u obliku **naziv_čvora / redni broj_sloga** u tablici izlaznih atributa navedenog nadređenog čvora. Tablica ulaznih atributa korijenskog čvora plana je prazan skup.

5.2.7. Izlazni atributi

Skup izlaznih atributa sadrži podatke o varijablama koje predaje akcijska funkcija nakon izvršenja, kao izlazne podatke. Skup izlaznih atributa u pravilu ne mora biti kompletan skup izlaznih podataka akcijske funkcije. U skupu izlaznih atributa čvora moraju se navesti oni izlazni podaci akcijske funkcije koji će biti potrebni podređenim čvorovima kao ulazni atributi ili podaci za odluke i ograničenja. Osim tih izlaznih podataka mogu se dalje navesti podaci koji su od prioritetnog značenja za opis konstrukcijskog rješenja zadatka odnosno podzadatka, što je ciljno stanje koje se treba postići planom. Naime pregled kompletnih skupova izlaznih podataka svih izvedenih akcijskih funkcija moguć je i korištenjem modula korisničkog sučelja.

Članovi skupa izlaznih atributa slijede kao niz slogova iza kjučne riječi $OUTPUT_SET_TABLE. Skup ulaznih atributa sastoji se od slijedećih elemenata:

TV, RB, SI, OPIS

gdje su:

TV	tip varijable (I-integer), (D-double float), (C-character string)
RB	redni broj varijable u tablici
SI	simbolička oznaka varijable
OPIS	pojam koji predstavlja varijabla

5.2.8. Skup ograničenja

U toku izrade plana mogu se unaprijed postavljati relacije i uvjeti ograničenja za pojedine varijable iz skupova ulaznih i izlaznih atributa. Zadani uvjeti ograničenja provjeravaju se nakon izvršenja akcijske funkcije. Ukoliko neki od uvjeta nije zadovoljen, korisnik će biti upozoren, uz mogućnost da prekine ili nastavi izvođenje plana.

Iza ključne riječi $CONSTRAINT_SET_TABLE. slijede uređeni nizovi slijedećeg oblika:

TV, AV, OP, DG, GG, OPIS

gdje su:

TV	tip varijable (I-integer), (D-double float), (C-character string)
AV	adresa varijable
OP	operator (<, <=, ==, >, >=, !=, range (interval))
DG	donja granica (konstanta ili adresa varijable)
GG	gornja granica (0 - za sve vrste operatora osim range)
OPIS	komentar, opis ograničenja

Provjeri ograničenja mogu se podvrgnuti varijable iz tablica izlaznih atributa nekog od nadređenih čvorova (po putanji do korijena). Adresa varijable zapisuje se u obliku **naziv_čvora / redni broj_sloga** u tablici izlaznih atributa navedenog nadređenog čvora. Ukoliko se radi o istom čvoru, tj. onom kojem tablica ograničenja pripada, naziv čvora u adresi može se ispustiti. Tablica ograničenja može biti i prazan skup.

6. Struktura ekspertnog CAE sustava

6.1. Struktura ekspertnog sustava

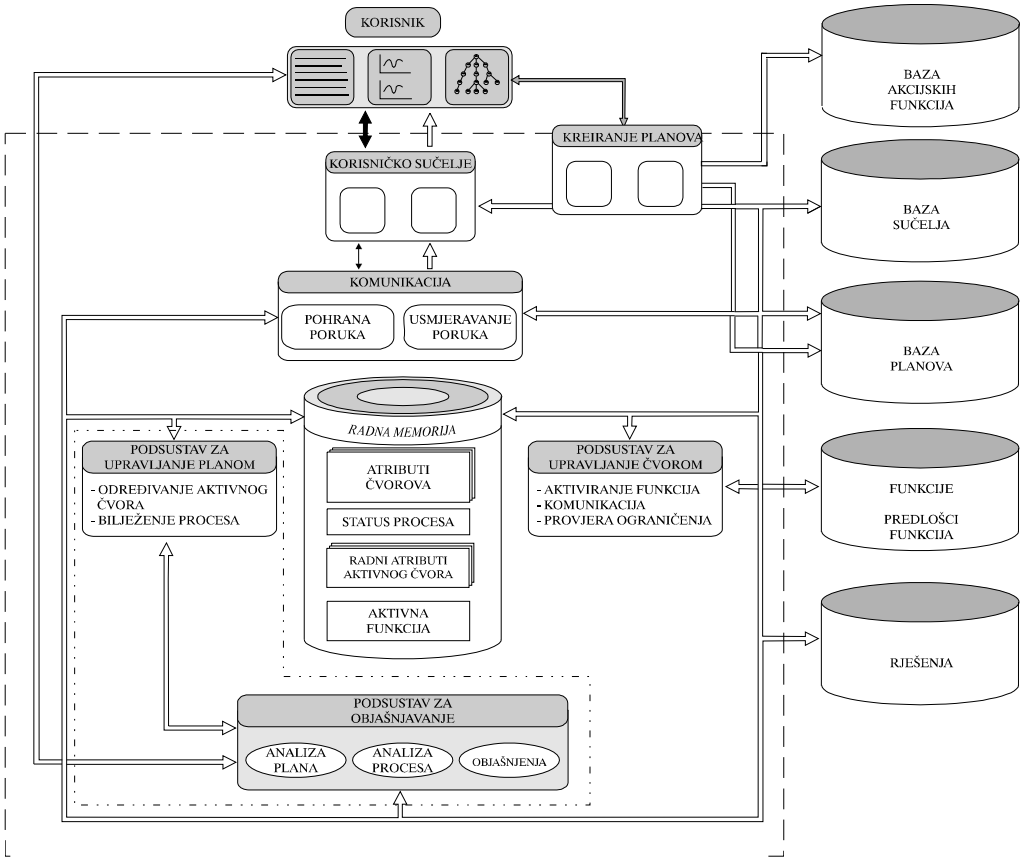
U ovom poglavlju dan je prikaz strukture cijelog ekspertnog sustava prema radovima i istraživanjima [9][7][17] i [27] koja su dijelovi zajedničkog projekta modeliranja i razrade ekspertnog, odnosno "inteligentnog" CAE sustava. Baza scenarija može funkcionirati kao samostalan modul, ali smisao svoje primjene nalazi tek kao element integriranog "ekspertnog" CAE sustava. Pri tome upotrebu baze scenarija možemo promatrati kao prvu fazu u procesu eksploatacije ekspertnog CAE sustava - odnosno fazu generiranja plana, nakon čega slijedi faza izvođenja plana u kojoj se koriste svi ostali moduli ekspertnog CAE sustava. Pri generiranju plana treba voditi računa i o tome kako se plan izvodi unutar okruženja ekspertnog CAE sustava. Struktura ekspertnog CAE sustava i zamišljeni algoritam izvođenja plana naravno imaju mnogostruke uplive i na način generiranja i strukturiranja plana. Način strukturiranja planova (u smislu podjele na podplanove i klasifikacije prema zadacima) može imati veliki utjecaj na efikasnost primjene cijelog ekspertnog CAE sustava u određenom okruženju.

Za sam proces izvođenja plana, ostalim modulima sustava potrebni su još neki skupovi podataka koji nisu dio zapisa plana, ali ih također treba definirati i zapisati u predviđene datoteke u tijeku generiranja plana.

Osnovu sustava čini prikaz procesa konstruiranja planom koji je opisan u trećem poglavlju. Pri tome struktura plana, sintaksa i semantika ne ovise o području primjene već o načinu implementacije modela. U ovom poglavlju biti će dani samo kratki prikazi podsustava koji su od temeljnog značenja za pokazivanje mogućnosti i načina korištenja zamišljenog modela ekspertnog CAE sustava.

Iz prikaza strukture ekspertnog CAE sustava (Slika 6-1) mogu se uočiti slijedeće funkcionalne cjeline (podsustavi):

- korisničko sučelje,
- podsustav za komunikaciju,
- podsustav za kreiranje plana (baza scenarija)
- podsustav za prevođenje plana,
- podsustav za upravljanje planom,
- podsustav za upravljanje stanjem čvora,
- podsustav za objašnjavanje.



Slika 6-1 Struktura ekspertnog CAE sustava

Treba napomenuti da je prikazani model strukture još uvijek u procesu razvoja, odnosno pojedine module treba detaljno razraditi, što može dovesti i do kasnijih manjih promjena u strukturi sustava.

Koncepcijom cijelog sustava kao i pojedinih podsustava određene su i neke od smjernica i zahtjeva za koncipiranje i realizaciju podsustava baze scenarija. To se posebno odnosi na podsustav za prevođenje plana čija je zadaća da prevodi plan iz simboličkog zapisa u binarni oblik, što znači da procedura za generiranje plana (u bazi scenarija) mora poštivati predviđenu sintaksu zapisa plana, odnosno osigurati sintaktički ispravan plan. Podsustav za prevođenje plana opisan je u [10] i [27].

6.1.1. Korisničko sučelje

Korisničko sučelje je za korisnika najviše eksponirani podsustav svakog programskog sustava, pa tako i ovog. Razvoj sučelja je sadržaj ovog rada, a dosadašnji rezultati na razvoju ovog korisničkog sučelja objavljeni su u radovima [26] [15] [16] [27]. Osnovni dijelovi korisničkog sučelja podijeljeni su na dvije razine koje čine:

- moduli za prikaze podataka i intrepretaciju podataka te komunikaciju s korisnikom (primanje poruka korisnika, te predaja poruka podsustavu za komunikaciju)
- modul za komunikaciju sa sustavom zadužen je za komunikaciju između korisničkog sučelja i podsustava za komunikaciju

Baza sučelja modula sadrži definicije raspoloživih sučelja pojedinih modula (alata). Tijekom rada, ovisno o raspoloživim podacima i trenutnoj funkciji modula, iz baze sučelja se na temelju poruke podsustava za upravljanje stanjem čvora aktivira sučelje potrebno za rad korisnika.

6.1.2. Podsustav za komunikaciju

Podsustav za komunikaciju osigurava pohranu i razmjenu informacija između svih ostalih elemenata sustava. To uključuje i kontrolu prioriteta razmjene informacija, te raspodjelu informacija od izvora poruke do cilja. Svi podsustavi ekspertnog CAD sustava razmjenjuju informacije porukama kroz ovaj podsustav. Koncepcijski se ovaj podsustav sastoji od modula za prihvat i raspodjelu poruka, te modula za pohranu poruka. Predloženi model grafičkog korisničkog sučelja opisan je u glavi 6, prototip realiziran na osnovu predloženog modela prikazan je u glavi 7.

6.1.3. Podsustav za upravljanje stanjem čvora

Ovaj podsustav služi za komunikaciju između podsustava za upravljanje planom i programskog alata, tj. akcijske funkcije koju treba aktivirati u čvoru. Zadaće su ovog podsustava :

- manipulacija porukama,
- aktiviranje predloška s opisom programskog alata (akcijske funkcije),
- generiranje informacija za korisničko sučelje,
- transformacija atributa čvora u ulazne podatke potrebne za aktiviranje funkcije,
- aktiviranje funkcije,
- provjera ograničenja i konzultacija korisnika ako nisu zadovoljena,
- transformacija izlaznih podataka akcijske funkcije u ulazne podatke čvorova sljedbenika.

Predložak s opisom programskog alata (akcijske funkcije) omogućuje relaksaciju krute forme atributa čvora, a zamišljen je i kao baza podataka koji su potrebni za aktiviranje određenog programskog alata. Takovi predlošci zapravo trebaju biti dio baze akcijskih funkcija i trebaju se popunjavati u isto vrijeme kad i sami skupovi atributa akcijskih funkcija.. Koncepcijom cijelog sustava predviđeno je da se kao akcijske funkcije mogu upotrebljavati bilo kakve programske aplikacije što implicira veliku raznorodnost načina izvođenja te učitavanja i ispisa podataka. Zbog toga je definiranje unificiranih predložaka za opis akcijske funkcije dosta zahtjevan zadatak, a većina smjernica koncipiranja proizlazi iz koncepcije podsustava za komunikaciju i podsustava korisničkog sučelja. U vrijeme dovršavanja ovog rada, nije još u potpunosti bila definirana struktura predložaka za opis načina izvođenja te učitavanja ulaznih i ispisa izlaznih podataka za akcijske funkcije, pa iz tog razloga što nisu uključeni u bazu akcijskih funkcija.

6.1.4. Podsustav za upravljanje planom

Ovaj podsustav upravlja korištenjem plana i predstavlja središnji dio sustava za korištenje plana. Rad ovog podsustava temelji se na informacijama sadržanim u planu i porukama koje generiraju podsustav za upravljanje čvorom, te korisnik. Nadzor integriteta podataka važna je zadaća ovog podsustava. Naime, ovdje se ustanovljuju greške ili neodređenosti u akcijama ostalih izvršnih članova sustava: korisnika i podsustava čvora. Greškom korisnika smatrati će se zahtjev za aktiviranjem čvora za kojeg vrijednosti ulaznih atributa nisu određene. U tom slučaju čvor se neće aktivirati, već će se konzultirati korisnik. Greškom podsustava čvora smatramo situaciju kada iz poruke podsustava čvora proizlazi da funkcija nije obavljena, o čemu će se također proslijediti poruka korisniku. Algoritam upravljanja planom prikazan je u [10][27].

6.1.5. Podsustav za objašnjavanje

Objašnjenje procesa konstruiranja, odnosno objašnjavanje izvršenih akcija od bitnog je značenja za kvalitetu eksploatacije plana i sigurnost korisnika. Podsustav za objašnjavanje koncipiran je kao ekspertni sustav, a detaljno je razrađen i opisan u [9][11]. Objašnjavanje se promatra kao problem rješavanja složenog zadatka prezentacije izvedenog procesa konstruiranja. Osnova ovog pristupa proizlazi iz koncepcije prikaza procesa konstruiranja planom koji služi kao okolina za eksploataciju različitih programskih alata koji, pojedinačno, služe za rješavanje parcijalnih konstrukcijskih zadataka u okviru cjelokupnog zadatka. Stoga je zadatak sustava za objašnjavanje interpretacija odvijanja konstrukcijskog procesa, s iznošenjem parametara odlučivanja koji su korišteni tijekom procesa.

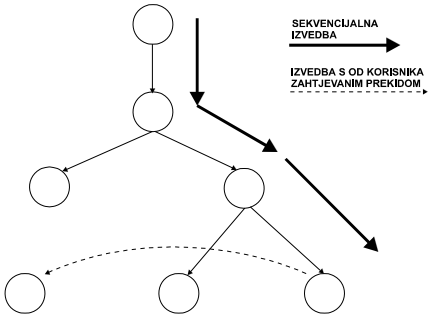
6.2. Fenomenološki opis rada sustava

U ovom poglavlju dan je prikaz rada sustava u fazi izvođenja plana, prema [9]. Treba naglasiti da u procesu kreiranja plana stalno treba voditi računa o načinu (algoritmu) izvođenja plana, odnosno, o načinu rada cjelokupnog sustava. U procesu kreiranja trebalo bi pokušati strukturirati plan na taj način da se u procesu izvođenja mogu eventualno prebroditi i neke od planom nepredviđenih situacija. Neke od takvih nepredviđenih situacija vjerojatno bi se mogle rješavati prijelazom na prekidani način izvođenja plana, a upravo struktura stabla čvorova plana, te struktura skupova ulaznih i izlaznih atributa čvorova mogu u takvim situacijama biti od velikog značaja.

Izbor plana koji će se aktivirati, korisnik vrši unutar korisničkog sučelja. Korisnik može pokrenuti izvođenje odabranog plana u cjelini, ili samo za neke čvorove. Nakon odabira plana, upravljanje preuzima podsustav za upravljanje planom (podsustav plana).

Za svaki čvor plana koji je na redu za aktiviranje, podsustav plana predaje upravljanje podsustavu za upravljanje čvorom (podsustav čvora).

Postoje dva temeljno različita načina izvođenja: sekvencijalni i prekidani (Slika 6-2).



Slika 6-2 : Sekvencijalni i prekidani način rada

Sekvencijalna izvedba znači da se plan izvodi kao cjelina, po putanjama određenim relacijama između čvorova. Upravljanje je potpuno pod nadzorom podsustava plana i čvora. Izvođenje s prekidom događa se kada korisnik kroz korisničko sučelje prekine sekvencijalnu izvedbu, te indicira koji čvor želi aktivirati. Korisnik može mijenjati načine izvođenja po svojoj želji.

Ovdje treba naglasiti da mogućnost prekidanog izvođenja plana ima veliki utjecaj na moguće načine promišljanja i koncipiranja u tijeku generiranja plana. Prekidanim načinom izvođenja može se u nekim slučajevima simulirati postupak iteracije, ako se to strukturom čvorova i atributa plana predvidi u tijeku generiranja plana. Prelaz iz sekvencijalnog u prekidani način rada vjerojatno bi se najčešće dešavao u situacijama kad nije zadovoljen niti jedan od uvjeta u tablici odlučivanja ili kada nisu zadovoljena ograničenja. Predviđanje takovih situacija također utječe na strukturiranje plana u tijeku generiranja, a može se pretpostaviti da to može biti i ključno za efikasnost i "eksploatabilnost" kreiranog plana. Takvo promišljanje mogli bi naznačiti i kao strukturiranje plana na taj način da bi se u tijeku izvođenja u slučaju potrebe mogao na eventualno predviđeni način simulirati prikaz konstrukcijskog procesa mrežom čvorova, pomoću prekidanog načina rada.

Čvor se aktivira (izvršava se akcijska funkcija) ako su određene vrijednosti ulaznih atributa čvora. Na ovoj razini provodi se i provjera ograničenja atributa za koje su postavljena.

Nakon provjere nadzor nad upravljanjem preuzima podsustav plana. Korisnik se obaviještava o daljnoj obradi, a u slučaju greške upravljanje se predaje korisniku kroz korisničko sučelje.

U slučaju prekida podsustav plana mora odrediti nastalu situaciju u odnosu na do sada prijeđeni put. U slučaju sekvencijalnog prekida (skoka) podsustav plana pohranjuje trenutno stanje, odnosno značajke provedenog procesa do trenutka prekida (tj. putanju i vrijednosti atributa aktiviranih čvorova). Razmatraju se dvije moguće situacije: prvi pokušaj aktiviranja čvora i bilo koji drugi pokušaj.

U potonjem se slučaju restauriraju vrijednosti atributa čvora u posljednjem pokušaju. Određuje se status izvođenja i vrijednosti atributa.

Ako su vrijednosti atributa određene, zahtijeva se potvrda korisnika za aktiviranje funkcije. Upravljanje se prepušta podsustavu čvora s odgovarajućom porukom.

Ako vrijednost nekog (ili nekih) ulaznog atributa nije određena, traži se od korisnika da odredi vrijednost atributa. Proces se nastavlja kao i u prijašnjem slučaju.

Proces završava kada korisnik to odredi. U slučaju sekvencijalnog izvođenja podsustav plana prekida izvođenje (predaje upravljanje korisniku) u slijedećim situacijama :

- kada se dostigne krajnji čvor,
- kada je vrijednost atributa izvan područja ograničenja,
- kada nije u stanju jednoznačno odrediti smjer grananja.

Ovdje ćemo navesti neke od mogućih grešaka koje se mogu napraviti u tijeku kreiranja plana, a da njihove posljedice budu uočene tek u tijeku izvođenja plana (ako uzrokuju neku od situacija prekida) :

- pogrešno upisani atributi u bazi akcijskih funkcija,
- pogrešno adresirani podaci (veze između atributa čvorova) u tijeku generiranja plana,
- pogrešno upisani podaci u predlošcima za opis akcijskih funkcija.

Na žalost uvijek postoji i mogućnost da neke od spomenutih grešaka ne budu uočene niti u tijeku izvođenja plana (ako ne uzrokuju situaciju prekida ili “nemoguće” izlazne rezultate). Stoga o tome treba posebno voditi računa pri testiranju kreiranih planova, u čemu veliki doprinos može imati podsustav za objašnjavanje.

7. Grafička korisnička sučelja

Tijekom stoljeća konstruktori su koristili prototipove nad kojima su obavljali eksperimente [23]. Provedeni eksperimenti su im ukazali na dobre i loše strane njihove konstrukcije. Na osnovu tih podataka donosili su zaključke o valjanosti konstrukcije i o mogućim poboljšanjima te se ponovo vraćali za "crtaću dasku". Ovakova metoda konstruiranja traje dugo što se pokazalo preskupim i presporim (zakonitosti tržišta [18]). Jedna od mogućih zamijena za prototipove su CAD aplikacije koje omogućuju konstruktoru vizualizaciju dijelova ili čitave konstrukcije, te omogućuju ograničenu iterativnost.

Bilo bi idealno kada bi CAD programi mogli pokriti sve faze procesa konstruiranja, ali to nije slučaj dijelom i stoga što postojeće CAD programske aplikacije trebaju vrlo točan opis objekata nad kojima se radi. Tako da je obrada informacija koje se pojavljuju u konceptulanoj fazi [18] procesa konstruiranja otežana. Tako da se CAD programske aplikacije koriste uglavnom kao razvojni alati samostalno ili u okviru nekog ekspertnog sustava [28][29][30].

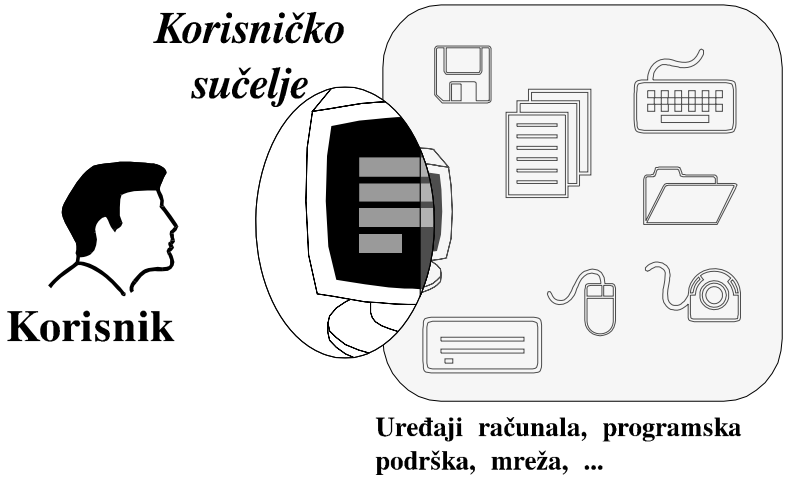
No bilo da se radi o CAD programskim aplikacijama, ekspertnim sustavima ili drugim programskim aplikacijama, svi imaju na neki način ostvarenu komunikaciju između računala i korisnika.

Kako su ljudske mogućnosti percepcije ograničene [23][31], a moderna računalska tehnologija nam omogućuje obradu ogromnih količina podataka uporabom

velike količine funkcija i pristupa, sve veći zahtijevi se postavljaju na korisnička sučelja i njihovu ulogu u komunikaciji između čovjeka i računala.

Korisničko sučelje sačinjava skup znakova, zvukova, grafičkih simbola te naredbi pomoću kojih korisnik i računalo komuniciraju. U početcima računalstva korisnička sučelja su bila relativno jednostavna, tekstualni ulaz i izlaz vrlo niska razina "interaktivnosti" prema korisniku. Snaga i kvaliteta današnjih računala omogućila je kvalitetniju komunikaciju korisnik - računalo, nove ulazno izlazne uređaje, te profinjenije programske aplikacije. Za očekivati je da će buduća korisnička sučelja podržavati govorni jezik, animirane trodimenzionalne prikaze te "inteligentne" module koji će automatski izvršavati određene zadatke u korist korisnika.

Svrha korisničkog sučelja je da ostvari i pojednostavi komunikaciju između korisnika i računala, integrirajući programsku podršku, dijelove računala i posebnosti pojednih aplikacija u dijalog (Slika 7-1). Ovdje se pod dijalogom smatra oblik komunikacije između korisnika i računala putem prozora na kojem se nalaze grafički objekti [32] kao nosioci informacije. Ovaj dialog skriva pravu strukturu ulazno/izlaznih uređaja, upravljačkog sustava, mreže i same aplikacije te omogućuje korisniku da upotrebljava aplikaciju neopterećen tehničkim mehanizmima u pozadini. Dakle korisničko sučelje stvara takav privid da korisnik ima "osjećaj" da može upravljati ne samo radom programske aplikacije nego i radom čitavog upravljačkog sustava, te da ga može, kao i programsku aplikaciju, prilagoditi svojim potrebama i željama.



Slika 7-1 Uloga korisničkog sučelja

Poznavanje i proučavanje komunikacije među ljudima je od velikog značaja prilikom koncipiranja korisničkog sučelja. Dijalog među ljudima se sastoji od simbola, govora te općepoznatog i priznatog skupa pretpostavki i znanja. Slično ljudskoj i

komunikacija čovjek - računalo se sastoji od zajedničkog znanja, te različitih rječnika i simbola.

Neovisno o razini razvijenosti programske podrške i dijelova računala, korisničko sučelje ostaje nepromjenjeno u svojim osnovnim funkcijama: jedan ili više oblika prikaza ideje, konceptualna organizacija podataka i funkcija, tehnike korištenja korisničkog sučelja (navigacije), karakteristike prikaza i slijed komuniciranja.

Prema [33] možemo razlučiti nekoliko osnovnih razloga koji dovode do loše realizacije korisničkog sučelja:

- kreatori grafičkih korisničkih sučelja najčešće kreiraju korisničko sučelje "za sebe" tj. prema svojoj predodžbi o korisničkom sučelju,
- kreiranje grafičkih korisničkih sučelja naizgled je jednostavan proces,
- kreatori korisničkih sučelja imaju u mnogome više znanja i iskustva u radu s korisničkim sučeljima nego oni koji će ih upotrebljavati,
- problem prikaza znanja,
- kreatori grafičkih korisničkih sučelja slabo poznaju ergonomiju,
- ignoriranje socialnih, kulturnih i psiholoških aspekata prilikom kreiranja grafičkih korisničkih sučelja,
- određeni principi kreiranja sučelja postaju samo parole, a ne koriste se,
- "ljudski faktor" je postala izlika za loš dizajn korisničkih sučelja.

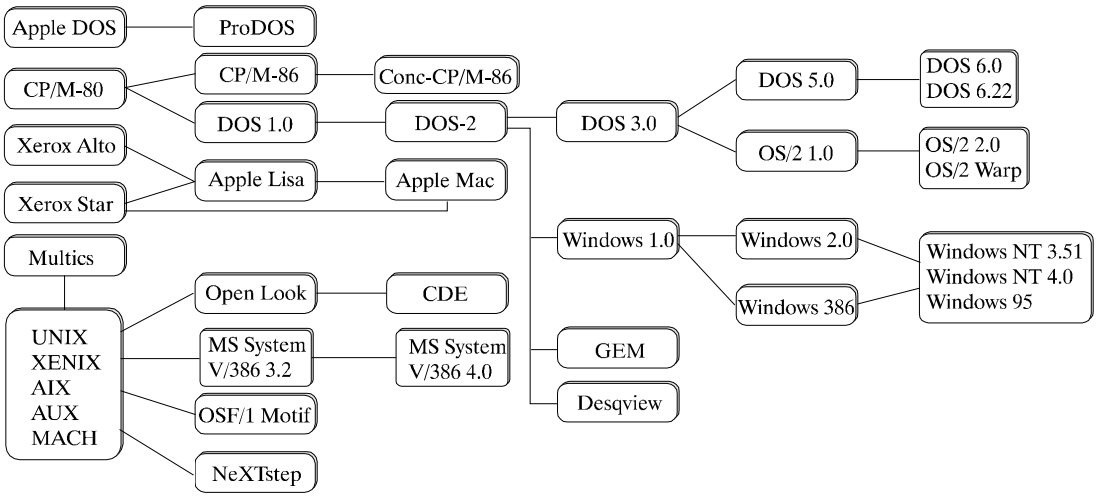
7.1. Pregled razvoja grafičkih korisničkih sučelja

Prema [32] razvoj korisničkog sučelja (Slika 7-2) prožet je revolucionarnim unapređenjima i periodima polagane promijene pa čak i zastoja u razvoju.

Grafička korisnička sučelja nisu nova ideja. Prvi put su opisana 1945. godine u članku koji je napiso Vannevar Bush. 1960. godine Ivan Sutherland kreirao je Sketchpad aplikaciju kao svoj diplomski rad. 1981. godine tvrtka Xerox je kao rezultat istraživanja provedenih tijekom 70tih u Palo Alto Research Centar ponudila Star sustav prozora.

Prvi sustav koji je ostvario kvalitetniji oblik komunikacije između računala i korisnika bio je RAND kreiran 1967. godine u RAND laboratorijima. Nakon što je Steve Jobs posjetio tvrtku Xerox pojavio se Apple Macintosh. Grafičko korisničko sučelje na Apple Macintosh računalu imalo je gotovo sve osobine kao i današnja grafička korisnička sučelja.

U sustavu prozora računala Apple Macintosh postoje neki elementi RAND sustava (promijenu geometrije prozora, automatsko ispravljanje geometrije itd.).



Slika 7-2 Pregled razvoja korisničkih sučelja

Do 1983. godine svaki veći proizvođač radnih stanica imao je svoj sustav prozora. 1984. godine na MIT (Massachusetts Institute of Technology) na projektu pod nazivom Athena razvijen je X-Windows sustav prozora. Ovaj sustav je napravio korak dalje te je omogućio korištenje aplikacija s drugih računala, putem mreže, na lokalnom računalu. 1985. godine tvrtka Microsoft ponudila je Microsoft Windows sustav prozora za PC (Personal Commputer) računala, tvrtka Digital Research GEM (za Atari i PC računala), tvrtka Comodore Amiga Intuition Interface sustav prozora, a tvrtka Apple sustav Lisa.

U [34] istaknuto je da su prije nastanka grafičkih korisničkih sučelja korisnici često postajali “frustrirani” pri svojim prvim susretima sa računalima, jer je izgledalo da računala jedino omogućuju tipkanje, sortiranje i pretraživanje, ono što su korisnici željeli je manipulacija idejama, a ne samo riječima.

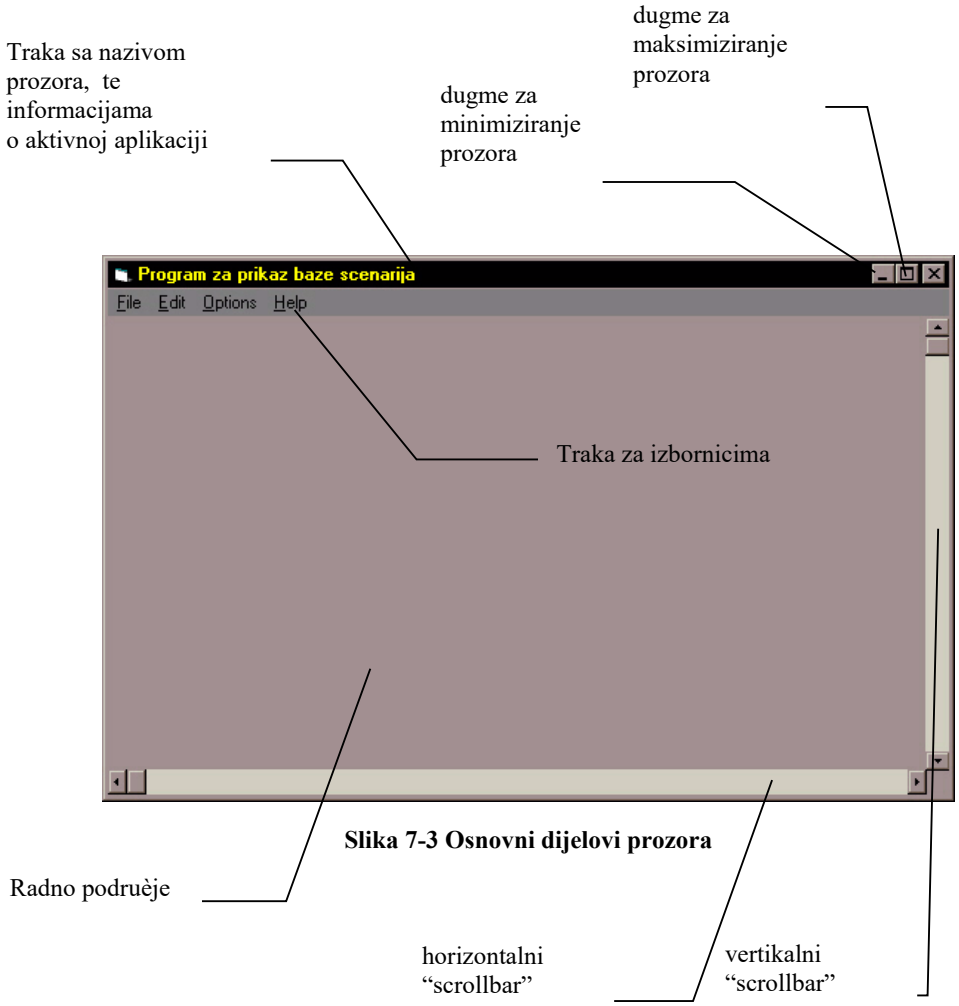
Tijekom 70tih i 80tih količina programskog koda potrebnog za ispis poruka sustava naglo se povećala. U to vrijeme, korisničko sučelje je bilo odvojeno od same aplikacije i postalo je predmet proučavanja programera i istraživača. Korisničko sučelje je danas toliko važno da je količina programskog koda utrošenog za izradu korisničkog sučelja mnogo veća od one utrošene za kodiranje same "poante" aplikacije.

Temeljem [32][34][35][33] tijekom protekla dva desetljeća, kreatori korisničkih sučelja uvidjeli su potrebu za istaživanjem i razvojem procesa i metoda potrebnih za izradu korisničkih sučelja. Korisnici najčešće preferiraju manju količinu funkcija (mogućnosti) uz dobro korisničko sučelje nego veliku količinu funkcija uz lošije korisničko sučelje. Atraktivnost u izgledu i lakoći korištenja korisničkog sučelja u velikom broju slučajeva je važnija od same funkcije aplikacije i njene izvedbe. (Pojam

izgled aplikacije, odnosi se na fizički prikaz korisničkog sučelja, a lakoća korištenja na karakteristike mehanizma korištenja sučelja). Dobro korisničko sučelje je između ostalog i proizvod programske strategije praćenja razvoja računalske tehnike.

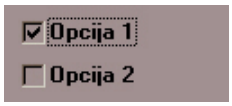
7.2. Dijelovi grafičkih korisničkih sučelja

U komercijalnim aplikacijama koriste se mnogi standardni elementi za izradu grafičkog korisničkog sučelja. Većina tih elemenata ima isti ili sličan naziv i funkciju neovisno o računalnoj platformi, upravljačkom sustavu ili upravitelju prozora koji se koristi. Na sljedećim slikama prikazani su i opisni osnovni i najčešće korišteni elemenati za izradu grafičkog korisničkog sučelja.

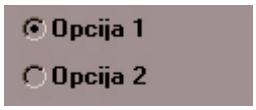


Slika 7-3 Osnovni dijelovi prozora

dugme - služi za ugrađivanje naredbi u dijalog te pokre-tanje aplikacija i drugih modula,



”**check box**” - služi za zadavanja jedne ili više opcija,

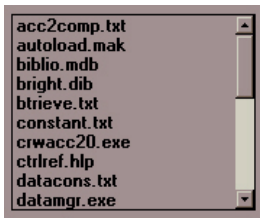


“**radio button**” - za određivanje jedne od ponuđenih opcija

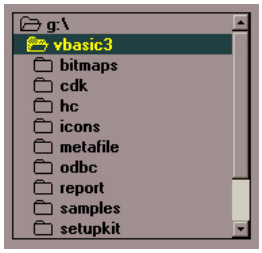
”**combo box**” - prikaz jednog elementa iz liste, oda-birom strelice može se dobiti ispis ostalih elemenata

ikona - grafički prikaz objekata (dijelova računala, logičkih cjelina, ili može služiti kao “ukras” na prozoru),

label - ispis teksta, ispisani tekst se ne može mijenjati osim kroz programsku aplikaciju



“**list box**”- prikaz više opcija ili logički povezanog skupa podataka (stablo direktorija, popis referenci itd.), kroz elemente liste može se kretati korištenjem vertikalnog “scrollbara”,



“**list box**” - opcije mogu biti tekstualne, ali tekstu mogu biti dodani i grafički simboli kao npr. ikone

okvir - služi za logičko organiziranje, grupiranje ili isticanje grupe grafičkih objekata,

“**panel**” - služi kao nositelj drugih grafičkih objekata te prikaz grafičkih objekata kao što su linije, kružnice itd.,

“**text box**” - služi za unos i ispis teksta, ispisani tekst se može mijenjati od strane korisnika ili aplikacije.

7.3. Komunikacija računalo-korisnik

Problemom komunikacije između računala i čovjeka bave se znanstvenici raznih profila: psiholozi, inžinjeri, računalni stručnjaci, stučnjaci za ljudsko ponašanja itd. Cilj istraživanja je ostvarenje grafičkog korisničkog sučelja koje je jednostavno za uporabu.

U [32] navode se četiri pristupa komuniciranju računalo-korisnik:

- iskustveni, osnova ovog pristupa je provođenje eksperimenata na sustava koji simulira moguću stvarnu situaciju,
- spoznajni, primjena teorije i iskustava iz psihologije i spoznajnih znanosti,
- predviđanjem, pokušaj predviđanja načina komuniciranja između korisnika i računala,
- antropološki, modela komuniciranja među ljudima korisi se kao model komuniciranja između čovjeka i računala .

Iskustveni pristup

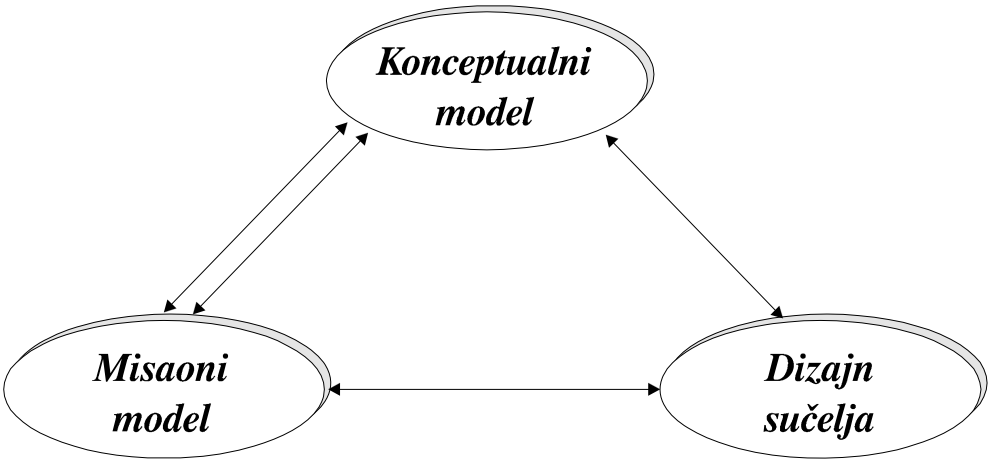
Kreatori grafičkih korisničkih sučelja imaju na rasplaganji različite "ulazne" uređaja: miš, tastatura, mikrofon, "touch" zaslonu itd. Koji će se uređaj korisiti, temeljem iskustvenog pristupa, ovisi o rezultatima provedenog eksperimenta. Metodologija ovog pristupa bliska je metodama koje se koriste pri proučavanju ljudskog ponašanja. Prvo se prepozna objekt proučavanja. Zatim se kreira model sustava koji se može jednostavno nadzirati, a koji je vjerna kopija stvarne situacije. Nakon toga se provede eksperiment, a rezultati dobiveni putem eksperimenta se analiziraju.

Koristeći se iskustvenim pristupom kreatori korisničkih sučelja moraju analizirati rezultate dobivene ekperimentalno. Provoditelj eksperimenta mora voditi računa da se rezultati dobiveni eksperimentom mogu primjenti i na duge slične situacije. Baza podataka kreirana na osnovu rezultata dobivenih eksperimentom mora se pohraniti i sačuvati radi daljnjeg proučavanja.

Prednost korištenja iskustenog pristupa je u tome da ovaj pristup nudi alternativu intuitivnom kreiranju. Budući da ljudi koriste računala, mehanizmi koji se čine najjednostavnijim za korisnike, moraju biti korišteni prilikom kreiranja grafičkih korisničkih sučelja.

Spoznajni pristup

U svrhu pojednostavljenja i povećanja efikasnosti u komunikaciji između računala i čovjeka korištenjem spoznajnog pristupa (Slika 7-4) primjenjena su iskustva iz psihologije. Spoznajne teorije daju nam informacije o tome kako čovjek koristi trajnu i privremenu memoriju pri manipulaciji informacijama, donošenju odluka i rješavanju problema.



Slika 7-4 Prikaz modela spoznajnog pristup.

Sa slike možemo razlučiti osnovne dijelove modela spoznajnog pristupa:

- **konceptualni** model je opis računalnog sustava korištenjem inžinjerske terminologije, tako da je opis točan, potpun i dosljedan,
- **misaoni** model je korisnikov (ljudski) opis načina rada računala, misaoni model upravalja korisnikovim akcijama,
- **dizajn sučelja** uključuje realizaciju korisničkog sučelja i izradu potrebne dokumentacije.

Kreator korisničkog sučelja temeljem spoznajnog pristupa mora razumjeti način promišljanja potencijalnog korisnika da bi mogao izraditi jednostavno i efikasno korisničko sučelje.

Pretpostavlja se da je korisnik prilagodljiv i fleksibilan, te da je važan čimbenik koji je aktivno uključen u proces kreiranja korisničkog sučelja. U spoznajnom se pristupu koriste: teorija o analognom zaključivanju i metafori, teorija o parcijalnom zaključivanju, teorija o problemu rješavanja problema i teorija neuralnih mreža.

Pristup predviđanjem

Svrha ovog pristupa je pokušaj predviđanja načina čovjekovog ophođenja sa računalima. Princip korišten u ovom pristupu je sličan onome koji se koristi pri kreiranju prototipova. Ukoliko se pri koncipiranju grafičkog korisničkog sučelja koristi pristup predviđanjem može se odrediti da li je predviđeni model ostvariv prije same izvedbe.

Antropološki pristup

Pojam antroplološki se može definirati kao primjena ljudskih osobina na uređaje. U okvirima ovog pristupa kreator korisničkog sučelja koristi model komuniciranja između ljudi kao model komuniciranja između računala i čovjeka. Ljudi mogu u većini slučajeva uspješno komunicirati, tako da osnovni problem u komunikaciji između računala i čovjeka leži u činjenici da se računala ne mogu ponašati kao ljudi prilikom komunikacije računalo-čovjek. Iz čega slijedi, da bi se pospješila komunikacija između računala i čovjeka, računalo mora poprimiti ispravne ljudske osobine.

Da bi se komunikacija između računala i čovjeka što više približila modelu komuniciranja između ljudi, računalima tj. programskim aplikacijama se dodaju razna poboljšanja: govorni jezik, komunikacija govorom, sustavi za učenje, efikasan sustav pomoći.

"Prilagodljiv" je izraz koji se koristi za opis jednostavnosti u korištenju programskih aplikacija. No, definicija samog termina nije jednostavna iz razloga što se izraz koristi za razne proizvode bilo programske aplikacije bilo uređaje računala tako da se pravi smisao izraza donekle izgubio. Izraz "prilagodljiv" u slučaju korisničkih sučelja se odnosi na lakoću i jednostavnost uporabe, "logički" slijed operacija, otvorenost, kratko vrijeme potrebno za savladavanje rada, te efikasan i obiman sustav pomoći.

Neki korisnici, prvenstveno iskusni korisnici programskih aplikacija u velikom broju slučajeva ne žele pretjerano "prilagodljiv" sustav. Iz tog razloga što pretjerano "prilagodljivo" korisničko sučelje tj. programski sustav može u pojedinim slučajevima usporiti rad iskusnog korisnika, te ga na taj način odbiti od uporabe programskog sustava ma kako on efikasan i korisan bio. Tako da se ispravno koncipirano korisničko sučelje treba moći prilagoditi "kvalificiranosti" tj. iskustvu potencijalnog korisnika.

Prednost antropološkog pristupa je što se ono može uporabiti pri koncipiranju i kreiranju potpuno novog korisničkog sučelja umjesto potrebe za slijeđenje drugih disciplina kao u iskustvenom pristupu.

U članku [36] opisana su dva modela za formaliziranje dijaloga između čovjeka i računala: jezični model i izvedbeni model.

Jezični model

U procesu komunikacije između računala i korisnika koriste se različiti jezici za različite smjerove komunikacije. Jedan jezik za komunikaciju korisnik-računalo i drugi za

komunikaciju računalo-korisnik. Prvi jezik, je ulazni jezik tj. jezik koji izražava aktivnosti korisnika npr. uporaba raznih vanjskih uređaja (tipkovnice ili miša). Izlazni jezik tj. jezik kojim računalo šalje korisniku različite poruke i podatke izražen je putem grafike na zaslonu te preko zvuka i simuliranog govora. Ova dva jezika imaju dvije glavne komponente: značenje (sadržaj) i oblik (kako je sadržaj prikazan) (Slika 7-5). Značenje se može podijeliti na konceptualno i funkcionalno, dok se oblik (izgled i osjećaj) može podijeliti u niz samostalnih interaktivnih dijelova koji su usko povezani sa fizičkim komponentama računala. Konceptualna komponenta je spoznajni model, tj. kako korisnik shavaća. Ova komponenta uključuje dijelove (objekte) sustava i njihova svojstva te odnose i funkcionalne operacije među objektima. Funkcionalni opis detalja, za svaku operaciju, prikazuje koje se greške mogu pojaviti te kako se one obrađuju.

Slika 7-5 Jezični model

Sintaktička razina određuje redosljed ulazno izlaznih operacija. Na leksičkoj razini ostvaruje se veza sa uređajima računala tj. određuje se kako se ulazne i izlazne jedinice (dijalozi, labele, ...) zapravo prikazuju ovisno o ugrađenim uređajima računala (hardware). U slučaju izlaza, ova razina određuje npr. kako će se prikazati crvena linija na ekranu, visinu tona na zvučniku ili razlučljivost zaslona. Za ulazne operacije, ova razina određuje tehnike komuniciranja sa korisnikom ovisno o raspoloživim uređajima npr. kontrola miša ili tableta. Sintaktička i leksička razina zapravo određuju izgled i "osjećaj" korisničkog sučelja, a time i same aplikacije.

Ilustrirat ćemo dosadašnje izlaganje na primjeru programa za crtanje geometrijskih oblika u ravnini. Na konceptualnoj razini, korisnik može kreirati, brisati i mijenjati geometrijske oblike na proizvoljno velikom području (crtežu), koristeći raspoložive alate

za pomoć (mreža točaka, ravnala, pomoćne crte). Operacije koje se mogu primjeniti na zasebnim geometrijskim oblicima uključuju: translaciju, rotaciju, povećavanje ili smanjivanje, brisanje, umnažanje te operacije promjene atributa geometrijskih oblika. Crtež se može spremiti, učitati te spremiti pod novim imenom. Funkcionalna razina daje detaljni opis objekata i operacija dok sintaktička razina određuje oblik komunikacije i njen format. Na primjer, da bi se obrisao objekt, korisnik ga mora prvo odabrati te zatim odabrati jedan od oblika zadavanja operacije brisanja. Na leksičkoj razini, brisanje se može prikazati kao "odvlačenje" objekta u "kantu za smeće", brisanje pritiskom na kombinaciju tipki ili odabirom naradbe za brisanje sa izbornika. Dakako, postoji velik broj mogućnosti i oblika zadavanja naredbi, koji će oblik biti stvarno implementirani ovisi o mogućnostima uređaja instaliranih na računalu, mogućnostima upravitelja prozora te potrebama same aplikacije i umješnosti kreatora.

Izvedbeni model

Izvedbeni model (Slika 7-6) pretpostavlja da se aplikacija koristi pod kontrolom nekog upravitelja prozora. Upravitelj prozora kontrolira sve ulazno izlaze operaciju unutar sustava i rad svih ostalih funkcija sustava. Kao takav može se smatrati produžetkom samog upravljačkog sustava. Upravitelj prozora sastoji se od dva dijela: samog upravitelja prozora, uz pomoć kojega korisnik otvara, zatvara i upravlja prozorima, i sustava prozora, koji kreira prozore i kontrolira njihove funkcije. U stvari upravitelj prozora je samo specijalni agent koji pomaže korisniku pri kontroli aplikacija.

Slika 7-6 Izvedbeni model

Da bi se moglo kreirati korisničko sučelje, kreatori prozora i upravitelja prozora određuju i isporučuju skup alata, biblioteka, gotovih rutina i dijelova korisničkih sučelja potrebnih za kreiranje korisničkih sučelja programskih aplikacija. Dijelovi grafičkih korisničkih sučelja uključuju dijaloge, forme, izbornike (padajuće i iskakajuće) i njihove komponente te pomičnu traku, ikone i trake s izbornicima. Skup alata (toolkit) omogućuje hierarijsku organizaciju djelova korisničkog sučelja koji se mogu opisati pomoću objektno orijentiranog pristupa, klasa, podklasa i mehanizma nasljeđivanja. Svaki dio grafičkog korisničkog sučelja je derivacija neke klase. Unutar svake klase definirane su metode tj. (funkcije) preko kojih derivirani objekti ostvaruju komunikaciju s drugim objektima deriviranih iz iste ili druge klase. U pojedinim slučajevima je potrebno prilagoditi karakteristike neke metode trenutnim potrebama. Promijena metoda se ostvaruje preko mehanizma preopterećenja. Metode koje su definirane unutar neke klase mogu se preopteretiti (overloading) [37], tj. metoda se preoptereti na taj način da se napiše vlastita metoda istog imena. U alatima za npr. X-Windowse većina objekata grafičkog korisničkog sučelja su derivacija osnovne bazne klase, primitive prozora.

Alati najčešće imaju procedure ponovnog poziva koje se aktiviraju prilikom pojave određenog događaja. Derivirani prozor najniže razine u hijerarhiji nasljeđivanja ("child window") može sam obrađivati događaje npr. pritisak na tipku miša ili ih može prosljediti prozoru više razine. Neke od ovih metoda su uključene u skup alata (npr. osvjetljenje određenog dugmeta i "resetiranje" ostalih) dok su druge, dio aplikacije i ovise o njenoj funkciji. Postojeći dijelovi grafičkog korisničkog sučelja tako mogu posjedovati postupke

za kontrolu događaja i postupke za izvršenje određene zadaće. Dijelovi korisničkog sučelja složeni iz dijelova s više hijerarhijske ljestvice mogu imati ograničenu interakciju. Na primjer, uključenje radio dugmeta na panelu će rezultirati resetiranjem prethodno uključenog radio dugmeta. Rijetko su različiti dijelovi korisničkog sučelja u interakciji samo preko postupaka koji su na raspolaganju iz alata (toolkit). Kodiranje je često potrebno ako se želi ostvariti međuovisnost različitih dijelova sučelja i kreiranje kompleksnijeg ponašanja. Na primjer, korisnik odabere neki objekt na prozoru i objekt se osvjetli, korisnik zatim odvlači objekt u "kantu za smeće" koja se također osvjetli te odpusti dugme na mišu i time obriše objekt. Ovakav slijed događaja, ili mora biti eksplicitno kodiran unaprijed, ili više razine upravljanja korisničkim sučeljem moraju omogućiti ostvarivanje takovog slijeda događaja.

Prema [38] korisnička sučelja se mogu podijeliti na osnovu funkcija koje nude u dvije kategorije: zadaćno orijentirana i sučelja s direktnom manipulacijom.

Kod **zadaćno orijentiranih** korisničkih sučelja korisniku su prikazane moguće akcije tj. zadaće koje sučelje može obaviti. Korisnik može odabrati jednu od ponuđenih akcija. Potom ga korisničko sučelja vodi kroz sve faze potrebene za izvođenje zadane akcije. Dobar primjer za ilustraciju izloženog je rad u “File Manageru” sa datotekama i direktorijima. Na izborniku “File Managera” korisniku su ponuđene naredbe za umnažanje, promijenu imena i brisanje datoteka i direktorija. Da bi započeo neku akciju na primjer brisanje datoteka ili direktorija korisnik uporabom miša (ili nekog drugog uređaja: tipkovnica, “light pen”) odabere dugme za brisanje ili sa izbornika odabere naredbu za brisanje. Nakon aktiviranja naredbe na ekranu će se pojaviti prozor sa listom datoteka i direktorija od kojih korisnik treba odabrati one nad kojim želi da se obavi akcija. Nakon odabira datoteka ili direktorija koje korisnik želi obrisati, upravljački sustav (“File Manager”) obriše odabrane objekte čime je zadaća završena.

Drugi način rada imaju korisnička sučelja s **direktnom manipulacijom**. U ovom načinu rada korisniku su na raspolaganju grafički objekti (simboli, slike, ...) koji predstavljaju elemente programskog sustava (datoteke, direktoriji, računala na mreži, ...) te ih on može odabrati. Nakon odabira grafičkih objekata korisnik bira operaciju koju želi obaviti nad odabranim objektima. Ako kao primjer, ponovo uzmemo “File Manager” i operaciju brisanja, korisnik prvo odabre datoteke i/ili direktorije koje želi obrisati te ih zatim “odvuče” na ikonu koja prestavlja objekt za brisanje (“kantu za smeće”). Korisničko sučelje zatim interpretira korisnikovu akciju te ukoliko se ona može obaviti nad odabranim objekti provede akciju tj. obiše odabrabe datoteke i/ili direktorije. Dakle akcija brisanja je ostvarena direktnom manipulacijom nad odabranim objektima.

Oba opisana korisnička sučelja na kraju obave istu zadaću na različite načine. Teško je razlučiti koji je od navedenih pristupa bolji, tako da većina modernih korisničkih sučelja podržavaju oba, a korisnicima je ostavljena mogućnost izbora.

7.4. Trend u razvoju korisnička sučelja

Prosječni korisnik računala, koji ima poteškoća i prilikom korištenja kotrola VCR-a (Video Cassette Recoreder), potencira potrebu za korisničko orijentiranim kreiranjem korisničkog sučelja. Korisničko orijentiran dizajn sučelja ne oslanja se na tehnologiju već na korisnikovu sposobnost zaključivanja, percepciju, izobrazbu, radnu okolinu i radno mjesto. Kreatori korisničkog sučelja tretiraju korisnika kao kompaktnu homogenu cjelinu koja se razlikuje samo po zadatku koji obavlja. Kako se broj i različitost korisnika povećava, a i računalni uređaji su sve bolje i potpunije opisani te mogu obrađivati sve veće količine podataka, došlo se do zaključka da se mora više pažnje posvetiti proučavanju korisnika, njegovih navika i potreba.

Današnja korisničko orijentirana sučelja imaju mogućnost prilagodbe izgleda i načina korištenja sučelja ovisno o korisnikovim zahtijevima i potrebama. Ova prilagodba se odnosi prvenstveno na promjenu rasporeda i vidljivost dijelova grafičkog korisničkog sučelja na prozoru, te promjenu boje i vrste slova, a u budućnosti korisnik će moći mijenjati govorni jezik za komuniciranje s aplikacijom te vizuelni indentitet same aplikacije. Tijekom sljedećih godine korisnička sučelja će postajati sve fleksibilnija, i omogućavati će prikaze modela različitih apstrakcija prikaza kako podataka tako i same funkcije aplikacije.

Korisnička sučelja su danas na visokom stupnju razvijenosti. Dijelovi korisničkih sučelja su većinom poznati i opisani. Zahtijevi na korisnička sučelja postavljaju velike izazove kreatorima korisničkog sučelja. Na primjer, jednostavnost i lakoća upotrebe se kadkada miješa sa jednostavnošću i lakoćom pristupa (koliko brzo mogu doći do tamo?), te lakoćom shavaćanja i razumjevanja (koliko dobro razumijem gdje sam stigao?). Kreiranje korisničko orijentiranih grafičkih sučelja je proces sličan onome prilikom pristupa problemima u arhitekturi i strojarstvu. Razvoj započinje analizom problema, koji dovodi do potrebe za intervjuiranjem korisnika te određivanjem potreba i kreiranjem plana uporabe. Četiri jezične razine (konceptualna, semantička, sintaktička i leksička) u skladu su s daljnjim koracima u razvoju korisničkog sučelja. Daljnji koraci u razvoju korisničkog sučelja najčešće nisu sljedni već uključuju izrade prototipa te višestruke iteracije [39].

Budući da se ne može zadovoljiti svim zahtijevima (jednostavnosat i lakoća uporabe, fleksibilnost, neovisnost o upravljačkom sustavu itd.) na korisničko sučelje

potrebni su neki ustupci. Lakoća upotrebe, ponekad nije osnovni kriterij koji se mora poštivati prilikom izrade korisničkog sučelja. Kreatori moraju korisničko sučelje prilagoditi potrebama korištenja od strane neiskusnog korisnika, ali se mora voditi računa i o potrebama iskusnog korisnika. Naime, iskusan korisnik u toku svog rada obrađuje znatno veće količine podataka i postavlja veće zahtijeva na aplikaciju, a time i na korisničko sučelje.

Osobitosti uporabe korisničkog sučelja od strane iskusnog korisnika i korisnika novaka umanjuju uticaj prirodnog napredovanja, ono što na prvi pogled izgleda umjetno, može postati normalno kako se konvencije mijenjaju prema uporabi aplikacije većim dijelom od strane pučanstva, a ne od specijaliziranih korisnika. Tijekom ovog prijelaza neki dijelovi i funkcije korisničkog sučelja podložne su promijenama dok druge ostaju ne promijenjene. Osnovna svojstva korisničkog sučelja koji ne podliježu promijenama su prema [39]:

Upravljanje. Korisnik mora imati osjećaj da on direktno upravljanje radom aplikacije.

Predvidljivost. Ovo svojstvo pretpostavlja npr. da će se iskakajući izbornik uvijek pojaviti sa iste strane kursora.

Interaktivnost. Korisničko sučelje mora biti u stanju obraditi kratke i jezgrovite upute od strane korisnika na taj način da je korisnik u uvjerenju da radi sa “inteligentnom” aplikacijom.

Za razliku od navedenih svojstava, ostala svojstva mogu varirati, osobito ona vezana za izgled i jednostavnost uporabe sučelja. Estetika grafičkog korisničkog sučelja se uveliko promijenila od nespretnih tekstualnih do današnjih modernih sučelja sa velikom količinom grafike i poboljšanom funkcionalnošću.

Tijekom zadnjih deset godina došlo je do ubrzanog razvoja raznolikih uporabljivih alata koji se koriste u izradi korisničkih sučelja. Ovi alati omogućili su ugradnju dvodimenzionalnih dijelova grafičkih korisničkih sučelja koji su postali standard u razvoju korisničkih sučelja. Alati imaju grafičke editore koji omogućuju kreatorima grafičkih korisničkih sučelja opisivanje i izradu cijelog sučelja bez potrebe za programiranjem tj. kodiranjem u nekom od programskih jezika. Krajnji korisnik može čak i mijenjati neka od svojstava korsničkog sučelja (boju, vrstu slova) u toku korištenja. Prilikom izrade korisničkih sučelja moguće je zadati neka djelomična ograničenja i na taj način ostvariti određeni stupanj automatiziranja izgleda sučelja. Izlaz grafičkih editora

najčešće je programski kod. Ovako generiran kod veže se sa ostalim dijelovima aplikacije, pomoću dodatnih alata ili ugrađenih mehanizama u grafički editor.

Sustav upravljanja korisničkim sučeljem UIMS (User Interface Management System) obuhvaća dijelove ("widgets") skupa alata ("toolkit") i omogućuje dodatne funkcije ponajprije za određivanje slijeda akcija prilikom komuniciranja korisnika i računala. Dodatno UIMS može omogućiti potpuno kreiranje izgleda ekrana, sustava pomoći, sustava poruka o greškama, definiranje makro instrukcija, "korak nazad" operacija i profiliranje korisnika. Neke verzije UIMS imaju alate za pomoć pri manipulaciji s podacima vezanim za aplikaciju. Dijelovi aplikacije tj. neki postupci mogu se kreirati u tijeku izrade samog korisničkog sučelja. Mehanizam UIMS ovisno o korisnikovim akcijama poziva određene postupke pretežito preko postupka ponovnog poziva ("callback functions"), te na taj način mijenja dijalog i određuje koje akcije korisnik može dalje koristiti. Korisnik i UIMS dijele kontrolu nad dijalogom, mada je u nekim verzijama UIMS kontrola korisničkog sučelja strogo u okvirima UIMS. Trend u razvoju grafičkih korisničkih sučelja je u podijeljenoj kontroli komunikacije između UIMS i korisnika. Na taj način korisničko sučelje može "inteligentnije" odgovarati na zahtijeve korisnika i može ga bolje voditi kroz aplikaciju. Na primjer, novije razvojne okoline grafičkih korisničkih sučelja omogućuju trenutnu provjeru ispravnosti rada i izgleda korisničkog sučelja što u mnogome olakšava kreiranje i kontrolu grafičkog korisničkog sučelja.

Različiti UIMS dozvoljavaju kreiranje dijaloga na različite načine. Neki imaju grafičke editore za kreiranje dijagrama toka podataka ili dijagrama stanja sustava (npr. Windows). Drugi posjeduju poseban programski jezik za opis dijaloga (npr. Motif, User Interface Language) ili su orijentirani obradi događaja. Loša strana takovih programskih jezika, koji opisuju ponašanje, je u velikoj količini broja stanja, uvjeta i korisnikovih akcija koji se mogu pojaviti u tipičnoj interaktivnoj aplikaciji. Upravljanje ovako velikom količinom informacija je kompleksan zadatak koji postavlja velike zahtijeve na alate za kreiranje korisničkog sučelja, te potrebu za mogućnošću detaljnog uvida u rad svih postupaka korištenih u izradi korisničkog sučelja te pregled makro instrukcija.

Većina komercijalnih sustava daje veću potporu i slobodu u kreiranju dijelova korisničkih sučelja nego kompoziciji slijeda akcija, što se tradicionalno ostvaruje kodiranjem u nekom od programskih jezika. Tako da su najsloženiji dijelovi grafičkog korisničkog sučelja najslabije podržani. Na Institutu za tehnologiju u saveznoj državi Georgiji (USA) razvijen je UIDE (User Inetrface Development Environment) sustav za kreiranje grafičkih korisničkih sučelja. Ovaj sustav koristi formalnu specifikaciju modela zamišljenog korisničkog sučelja, te hijerarhijsku strukturu klasa i objekata za opis modela

uz upotrebu svojstava i metoda objekata, te akcija koje se obavljaju nad objektima prije i poslije pojave određenog događaja. Formalna specifikacija modela zamišljenog korisničkog sučelja posjeduje dovoljno informacija za automatsko kreiranje korisničkog sučelja (ali ne svih njegovih dijelova, naročito ne onih vezanih za vezu sučelja i same aplikacije). Nakon izrade osnovnog izgleda korisničkog sučelja programer može korisničko sučelje dodatno prilagoditi zahtijevima ili potrebama budućih korisnika.

Razoj korisničkih sučelja od "spartanskih" tekstualnih do današnjih modernih grafičkih korisničkih sučelja moguć je i zbog velikog napretka u razvoju uređaja računala ("hardware"), bitmap grafike, miša i "dedicated" računala. Budućnost razvoja računala i računalnih uređaja obećava uporabu dodatnih oblika komunikacije, govorni jezik, prepoznavanje gesta, trodimenzionalnu animaciju u stvarnom vremenu, te veći skup dijelova za izradu korisničkog sučelja. Kombinacija kratkih vizualnih i akustičkih poruka pokazala se kao dobro rjšenje za poboljšanje shvaćanja i razumijevanja informacija te njihovog zapažanja. Mogućnosti korisničkih sučelja, na počecima razvoja računala, bila su ograničena mogućnostima uređaja računala, monitori niske razlučljivosti, grafičke kartice skromnih mogućnosti (CGA, EGA), neadekvatni upravljački uređaji, no današnja računalska tehnika je toliko razvijena ta praktički nema ograničenja u ralizaciji ili korištenju grafičkih korisničkih sučelja.

7.5. Trodimenzionalna korisnička sučelja

Sve veći broj današnjih aplikacija ima trodimenzionalna korisnička sučelja. Ovakva sučelja se najčešće koriste kod CAD/CAM aplikacija, znanstvenih prikaza trodimenzionalnih prirodnih pojava, za animaciju u izobrazbi, zabavu i za potrebe umjetnosti. Također postoji rastuća potreba za trodimenzionalnim prikazom informacija i podataka koji nisu u svojoj osnovi trodimenzionalni, pa čak nisu niti grafički objekti, kao što je prikaz strukture baze, nap. organizacije zaposlenih u nekoj tvrtki. Trodimenzionalna okolina omogućuje potpuno iskorištenje sposobnosti ljudskog mozga u percepciji prostornih problema, ali i postavlja znatne poteškoće dizjanerima sučelja i potencijalnim korisnicima. Pomicanje miša ili nekog drugog ulaznog uređaja odgovara prirodnom poimanju pokreta u koordinantoj ravnini radne okoline i dozvoljava odabir objekata, odabirom njegove predodžbe na ekranu. U trodimenzionalnoj okolini, zaslon prikazuje samo dvodimenzionalnu projekciju stvarnih objekata. Svi objekti ili njihovi dijelovi nisu uvijek vidljivi u svim pogledima, pa ipak korisnik mora na osnovu datih pogleda stvoriti predodžbu o prostornoj slici objekta i njegovim geometrijskim osobinama. Neki statički prikazi mogu biti nerazumljivi. U rješenju ovoga problema pomaže tehnika stereo pogleda, koja stvara privid dubine. No, da bi se omogućilo potpuno razumijevanje stanja objekata i njihovih međusobnih odnosa, potrebno je da se

korisnik može slobodno "kretati" između objekata, korištenjem sustava kamera i pogleda. Za prikaz kretanja u stvarnom vremenu, što je poželjno, potrebna su računala visokih performansi, te skupi zasloni i grafičke kartice.

Ne samo da je izlaz, tj. prikaz kompliciran već je i ulaz tj. unos podataka i odabir opcija i elemenata također složen. Zbog toga se za unos koristi "spaceball" ili Pelhemusova olovka (uređaj sličan olovci, koji pomoću elektromagnetne veze između tri odašiljača i tri prijemnika određuje trenutni položaj u prostoru). Korisnici se u pravilu teško snalaze prilikom korištenja ovih uređaja za odabir objekata iz trodimenzionalnog prostora. Zbog ovih razloga, a i zbog visoke cijene uređaja potrebnih za podršku rad u prostoru, većina korisnika se opredjeljuje za tradicionalno dvodimenzionalno korisničko sučelje, te odnos podataka i odabir objekata pomoću miša. Kretnje u dvodimenzionalnom prostoru zaslona mogu se preslikati u različite ravnine i poglede, na primjer u globalni koordinatni sustav, lokalni koordinatni sustav nekog objekta, te u abstraktan parametarski prostor uključujući i trodimenzionalni prostor (x i y koordinata leže u ravnini zaslona, a z koordinata je okomito na zaslon).

Dizajneri i kreatori trodimenzionalnih grafičkih korisničkih sučelja moraju ugraditi obimnu bazu tehnika za komuniciranje između korisnika i korisničkog sučelja. Trodimenzionalna grafička korisnička sučelja moraju imati ugrađene mehanizme rasčlandbe korisnikovih akcija, nagađajući na osnovu njih "što je korisnik htio?". Informacije koje nedostaju dobivaju se raznim načinima komparacije i projekcijama. Ostala stanja koja su jednostavana za obradu u dvodimenzionalnom prostoru uveliko se kompliciraju dodavanjem treće koordinate. Na primjer, odabir objekta u ravnini zahtijeva izračunavanje točke ili odabir objekta najbližeg pokazivaču. U trodimenzionalnom prostoru, ista operacija može zahtijevati izračunavanje skrivenih linije, trodimenzionalno odsjecanje ("clipping") ili neke druge operacije u prostoru.

7.6. Multimedija

Činjenica je da dok se kod nas još uvijek raspravlja o samoj jezičnoj konstrukciji naziva multimedija, u suvremenom svijetu ljudi je već obimno koriste čak i za obavljanje svakodnevnog rada (bankovni automati, koordinacija rada i upravljanje pri remontu postrojenja, ...). Vremena se mijenjaju tako da se uporaba multimedije i kod nas zahuktava i polako ali sigurno ulazi u sve pore računalstva. Na pitanje što je multimedija, u principu nije teško odgovoriti, naime samo ime znači "više medija", no multimedija je znatno više nego "više medija". Multimedijsko okruženje obuhvaća različite modalitete komunikacije. Programski sustavi koji podržavaju multimediju omogućuju komunikaciju između računala, računala i čovjeka, te komunikaciju između ljudi pomoću računala

slikama, grafičkim objekatima (ikone, dugmad, simboli, ...), zvukom (prethodno snimljen i pohranjen na disk ili preko mikrofona i zvučnika direktno), animiranim skvencama i video zapisom.

U nekim radovima [40][41] multimedija se naziva hipermedija ili interaktivna medija i to zbog toga što je tvrtka Comptons registrirala ime "multimedija". No neovisno o nazivu, u svim aplikacijama se radi o spajanju različitih medija za prezentaciju grafike, zvuka, teksta ili video zapisa u cjelinu koja se može obrađivati računalom.

Kako je poznato ljudi lakše uče i bolje pamte informacije prezentirane kombinacijom modaliteta komuniciranja. Tekstualna informacija je u velikom broju slučajeva preformalna, a i ograničena je vokabularom i gramatikom jezika na kojem je napisana. S druge strane, sliku je lakše pojmiti, a i može nositi veću količinu informacija nego sam tekst ("slika govori više nego 1000 riječi"). No i prikaz informacija slikom ima nedostataka, naime ljudi različitih kulturnih, socijalnih, vjerskih, etničkih i profesionalnih skupina istu sliku mogu protumačiti na različite načine.

Nagli razvoj računala omogućio je ekspanziju multimedijske tehnologije. Ključ multimedije je interaktivnost. Interaktivnost se ostvaruje pomoću perifernih dijelova računala kao što su: trackball, video kamere, mikrofoni, zvučne kartice, "touch screen". Multimedija se čvrsto oslanja na prodornost CD (Comapact Disc) tehnologije. Naime, CD tehnologija je relativno jeftina, a omogućava zapis velike količine podataka uz razmjerno brz pristup.

Početkom 90tih tvrtka Apple je predstavila svoju QuickTime programsku rutinu koja na razini upravljačkog sustava omogućava prikazivanje video zaposa na računalu u stvarnom vremenu. U kratkom vremenu QuickTime se proširio sa Macintosh računala na PC računala tj. na MS Windowse (pod nazivom Video for Windows).

Razvoj Video for Windows standarda pokreno je i razvoj niza drugih programskih standarda za manipulaciju velkim količinama podataka, kompresijom podataka i njihovu obradu (npr. JPEG, AVI, MPEG, RTV). Ovi standardi najčešće su ugrađeni u razne sustavske funkcije. Uporedo s razvojem standarda i "hardwarea" razvijala se i programska podrška. U početku su se za kreiranje multimedijskih aplikacija koristitili progamski jezici npr. C ili Pascal, a kontroliranje količine informacija predstavljalo je ozbiljan problem.

Jednostavnije kreiranje multimedijskih aplikacije započelo je pojavom HyperCard aplikacije tvrtke Apple. Ova aplikacija omogućila je "običnim" korisnicima kreiranje

multimedijskih baza podataka. Na osnovi HyperCarda razila se kasnije paleta programskih aplikacija poznatijih pod nazivom autorski sustavi (“authoring systems”). Autorski sustavi omogućuju povezivanje računala na programskoj razini s raznim periferijskim uređajima. (CD-ROM, video uređaji, mikrofoni, audo izlazi itd.) Od poznatijih autorskih sustava na tržištu se mogu pronaći programski paketi poput Macromind Directora, Toolbook Multimedie, ASC interactiva ili Authorwarea. Zajedničko svim ovim programskim paketima je da omogućuju istovremeno kreiranje interaktivnih baza podataka koje sadrže tekst, sliku, animaciju, video zapis i sl.

Možda jedna od naraširenijih uporaba multimedije je u komunikaciji i prezentiranju informacija putem globalne mreže Interneta. Pomoću Internet pretraživača mogu se na lokalnom računalu prikazati video isječci, tekst, animacije, grafike te reproducirati zvuk.

Činjenica je da danas na tržištu programskih aplikacija sve je više aplikacija koje potpuno ili dijelom podržavaju multimediju ili neki segment multimedije. Multimedijske enciklopedije, knjige, promotivni materijali već su normalna pojava, tako da je za očekivati da će multimedija ovladati načinom komunikacije između korisnika i računala u svakodnevnom radu (banke, pošte, trgovine, ...).

7.7. Grupni rad

U osnovnoj zamisli o uporabi računala, za jednim računalom radi jedan korisnik i izvodi jedan zadatak. U sljedećem koraku razvoja računala jedan korisnik izvodi više zadataka uz prikaz rada svake aplikacije u zasebnom prozoru. Sljedeći logičan korak je zajednički rad više korsnika na jednom zadatku uz pomoć programa za podršku grupnom radu (“grupware”), na ovom stupnju prirodnim se čini uporaba "multimedije" i "hypermedije". Korisničko sučelje za aplikacije koje podržavaju grupni rad mora omogućiti prikaz trenutno aktivnih korisnika i poslova koje obavljaju isti, i to razlikujući rad nad zajedničkom i rad nad osobnim dokumentima ili aplikacijama te kontrolu rada dislociranih korisnika, bilo vremenski bilo prostorno. Za ostvarivanje ideje grupnog rada poželjno je postojanje mreže (jer fizički više korisnika ne može raditi u isto vrijeme na jednom računalu). Globalna mreža Internet omogućava grupni rad korisnika koji su zemljopisno udaljeni, čak i na različitim kontinentima. Programski jezik Java [42][43] tvrtke Sun Microsystems uveliko pojednostavljuje realizaciju korisničkih sučelja za podršku grupnom radu.

7.8. "Inteligentni" posrednici

"Inteligentni" posrednici su pomoćni programi i programski sustav za vođenje korisnika i pomoć pri uporabi računala ili neke programske aplikacije. Danas se mogu susresti računala koja komuniciraju s korisnikom govorom. Ova komunikacija je još u razvojnoj fazi, naime računala imaju ograničen spektar operacija koje mogu izvršiti ("Microsoft Windows speech recognition system"). Govorne naredbe koje računalo može razumjeti i izvršiti svode se na jednostavne operacije tipa "otvori", "zatvori", "spremi" i sl. Programske aplikacije koje omogućuju zadavanje izvršenja složenih operacija govorom su rijetke i skupe.

Korisnik prilikom korištenja "inteligentnih" posrednika može odabrati posrednika s određenom osobenošću npr. određenog naglaska. U CNRI (Corporation for National Research Initiatives) knjižnici uveli su knjižničke "robote" (knowbots) tj. programe zadužene za sortiranje, analizu, održavanje i pretragu informacija u raznim elektroničkim knjižnicama putem mreže koji komuniciraju s korisnicima putem govornih naredbi.

Uporaba "inteligentnih" agenata je sve raširenija . U "mainfraim" računalima agenti se koriste za pronalaženje, dijagnosticiranje i otklanjanje kvarova u radu računala. Neke kompanije koriste agente za pronalaženje informacija po Internetu. "Inteligentni" agenti se mogu korisno uporabiti i u ekspertnim sustavima te u aplikacijama koje koriste metode umjetne inteligencije [44][39], za prijenos informacija, pretraživanje baza znanja ili neke druge zadatke.

7.9. Grafički standardi

7.9.1. Važnost standarda

Nakon mnogih problema primjećenih tijekom uporabe programa koji su ovisni o uređajima računala ustanovljena je komisija za standarde. Komisija je usvojila dvije važne odluke koje su uticale na razvoj grafičkih standarda:

- **Portabilnost**. Prenosivost ili portabilnost programskih aplikacija između različitih računalnih sustava jedna je od važnijih ako ne i najvažniji cilj grafičkih standarda.
- **Prikaz modela**. Komisija za standarde je naglasila da postoji potreba za standardizacijom rada sa pogledima (u CAD aplikacijama), rad sa prikazima modela i skupom funkcija za interakciju sa korisnikom zbog potrebe za povezivanjem prikaza modela i njegovog zapisa u bazi.

Kreiranje programskih aplikacija danas je nešto više nego samo realizacija dobre ideje. Današnje programske aplikacije rade u kompleksnom okruženju koje se sastoji od

različitih upravljačkih sustava i grafičkih korisničkih sučelja. Kako je kreiranje i razvijanje programskih aplikacija diktirano potrebama tržišta, mogućnost brze implementacije i jednostavne instalacije određene programske aplikacije na različite računalne platforme je imperativ kojeg se proizvođači pokušavaju pridržavati. U njihovim naporima za implementiranje programskih aplikacija na različite računalne sustava od velike pomoći su i dobro razrađeni i implementirani standardi.

Budući da je vrijeme razvoja standarda dugotrajno proizvođači moraju unaprijed (u tijeku razvoja standarda) odlučiti da li će ga se pridržavati ili ne.

7.9.2. Pregled grafički standarda

CORE-3D

CORE (Core Graphics Systems) [34] je standard određen kao portabilan standard neovisan o uređajima računala za 2D i 3D grafički ulaz i izlaz. Standard je baziran na dugogodišnjim iskustvima u radu sa grafičkim aplikacijama čiji je rad i implementacija ovisila o uređajima računala. CORE je određen kao grafički standard koji simulira mogućnosti grafičkih terminala i "pen" plotera. U standard su ugrađeni elementi koji pojednostavljuju razvoj programskih aplikacija, te mehanizmi za kreiranje jednostavnih 2D i 3D grafičkih objekata (linija, ispunjeni oblici, polilinija itd.) i mehanizmi za transformaciju koordinatnih sustava i manipulaciju atributima grafičkih objekata (boja, popuna, ...).

CORE standard inicijalno nije podržavao rad sa složenijim grafičkim oblicima kao što su kružnice i elipse, ali je u tu svrhu postojao tzv. "escape" mehanizam preko kojeg je korisnik mogao prisupiti određenim funkcijama pojedinih uređaja ("device specific functions"). Jedna od najjačih osobina CORE standarda je dijeljenje ("segmentation"). Osnovni grafički oblici su grupirani unutar struktura ("segments") i kao takovi su pohranjeni u memoriju računala, tako da se kasnije u slučaju potrebe mogu pozvati. Stoga programska aplikacija nije morala ponovo kreirati već kreirane grafičke objekte nego ih samo pozivati iz memorije.

CORE standard ima nekoliko nedostataka, a to su:

- podržavanje samo jednog izlaznog uređaja,
- nepodržavanje standardnih sučelja između programske aplikacije i biblioteke funkcija koje implementiraju standard,
- potreba za velikim modifikacijama programske aplikacije u slučaju implementacije na drugoj računalnoj platformi koja također podržava isti standard.

PHIGS, PHIGS+

PHIGS ("Programmers Hierarchical Interactive Graphics Standard") [45] standard određuje sustave za kontrolu definicije, promijenu i prikaz hijerarhijski ograniziranih grafičkih podataka. U standardu je dan funkcionalni opis mogućnosti sustava, uključujući određivanje unutarnje strukture, mehanizama promijene, funkcija prikaza i kontrole. PHIGS posjeduje grafičke alate za podršku rada na visoko zahtijevnim radnim stanicama.

Grafički objekti su definirani unutar PHIGS grafičke baze kao slijed elemenata, uključujući osnovne grafičke objekte, atribute i transformacije. Standard omogućuju gladak ("smooth") dinamički prikaz 3D objekata.

Od 1988. godine PHIGS je dio ANSI standarda. Tijekom završnih radova na zaključenju standarda predložena su neka poboljšanja koja su dio PHIGS+ standarda.

PHIGS+ standard proširuje krug primjene PHIGS standarda na podršku realističnom "renderingu" grafičkih objekata i dodaje funkcije za uporabu više izvora svjetla, poboljšanje kvalitete sjenčenja, rad sa NURBS (Non Uniform Rational Bezier Spline) krivuljama i površinama, uporabu "true color" i kompleksnih grafičkih objekata.

GKS

GKS (Graphical Kernal System) [34] standard je postao dio ANSI standarda 1985. godine. Standard je bio prvenstveno orijentiran na izradu 2D shema, no zahtijevi su rasli tako da je standard proširen sa GKS-3D standardom. GKS-3D standard [46] određuje prikaz 3D žičanih objekata, te omogućava korisniku uporabu 3D ulaznih uređaja, skivanje linija i površina. Standard ne podržava kontrolu izvora svjetla, "renderiranje", sjenčenje i uprabu tekstura. Zbog ograničenja koje ima GKS-3D razvijen je i usvojen 1994 godine. GKS-94 standard [47] koji podržava NDC (Normalized Device Coordinate), osnovne objekte (krivulje, označivače, područja, karaktere, slike i crteže), atribute, korisnički definirane poglede, opis integracije u sustav prozora, metafile CGM (Computer Graphics Metafile) zapis crteža i osnovne klase.

PEX

PEX je nastavak PHIGS i PHIGS+ standarda (PHIGS Extension) za X11 sustav prozora. Ovaj standard omogućuje korisnicima uporabu naprednih grafičkih mogućnosti na radnim stanicama uporabom standardnih PHIGS grafičkih sučelja. PEX je razvijen kao standard koji podržava široku paletu grafičkih uređaja i uporabu grafike na visokoj razini. PEX nije razvojni alat već definicija protokola.

XIE

XIE (X Windows System Imaginig Extension) je dodatak na X sustav prozora koji određuje zapisivanje, učitavanje i prikaz grafičkih objekata u CCITT komprimiranom formatu. Standard podržava kontrolu boje, promijenu veličine, izoštravanje prikaza, “gamma correction” i “dithering”.

VEX

VEX je video komplement XIE dodatka. Ovaj standard određuje transfer i prikazivanje video zapisa u stvarnom vremenu na X sustavu prozora te mehanizme za kontrolu vanjskih video uređaja (CD-player, VCR (Video Cassette Recorder)).

SIE

Tvrtka NCD (Network Computing Devices, Inc.) koja je proizvođač terminala za X sustav prozora razvila je programe za sažimanje i manipulaciju zapisa grafičkih prikaza. Razvijeni programi se sastoje od šest dodatnih poziva skupu biblioteka X sustava prozora nazvanih SIE (Simple Imaging Extension).

XIM

XIM (X Input Method) [34] je jedan od najvažnijih standarda. Standard uključuje metode za rad sa skupovima internacionalnih znakova i njihovog rasporeda na tipkovnici. XView sustav kreiran u Japanu, koji se koristi za razvoj programskih aplikacija pod OpenLook sustavom prozora je zasnovan na XIM standardu. “X Consortium” je Ovaj standard odredio kao osnovu za rješavanje problema internacionalizacije X sustava prozora.

Display PostScript

DisplayPostscript (DPS) je razvila tvrtka Adobe Systems kao jezik za opis sadržaja zaslona. Trenutno je DPS integriran u sustav prozora tvrtke DEC i u kompatibilnom prozvodu XPS tvrtke Sun Microsystems kao dio X11/NeWS sustava prozora. Ovaj standard koristi “de facto” istu tehnologiju kao i PostScript jezik za opis stranica i tiskanje. DPS nije pogodan za X sustav prozora iz tog razloga što bi u slučaju implementacije DPS u X sustav prozora morao biti i implementiran interpreter za DPS kao dio “Server” programa X sustava prozora, no neki dijelovi DPS i X sustav prozora koriste iste funkcije. DPS nije “javno dostupan” niti je neovisan o proizvođaču i uređajima računala na koje se implementira.

7.10. Pregled grafičkih korisničkih sučelja

7.10.1. DECwindows

Slika 7-7 Izgled DECwindows grafičkog korisničkog sučelja

Slika 7-8 Struktura DECwindows grafičkog korisničkog sučelja

7.10.2. Macintosh

Slika 7-9 Izgled Macintosh grafičkog korisničkog sučelja

Slika 7-10 Struktura Macintosh grafičkog korisničkog sučelja

7.10.3. Motif

Slika 7-11 Izgled Motif grafičkog korisničkog sučelja

Slika 7-12 Struktura Motif grafičkog korisničkog sučelja

7.10.4. NeXTstep

Slika 7-13 Izgled NeXTstep grafičkog korisničkog sučelja

7.10.5. X11/NeWS

Slika 7-14 Struktura NeXTstep grafičkog korisničkog sučelja

Slika 7-15 Izgled CDE (Common Desktop Environment) aplikacije na X11/NeWS grafičkog korisničkog sučelja

Slika 7-16 Struktura X11/NeWS grafičkog korisničkog sučelja

7.10.6. OPEN LOOK,OpenWindows

Slika 7-17 Izgled OpenWindows grafičkog korisničkog sučelja

Slika 7-18 Struktura OpenLook grafičkog korisničkog sučelja

Slika 7-19 Struktura OpenWindows grafičkog korisničkog sučelja

7.10.7. Presentation Manager

Slika 7-20 Izgled Presentation Manager grafičkog korisničkog sučelja

Slika 7-21 Struktura Presentation Manager grafičkog korisničkog sučelja

7.10.8. Microsoft Windows

Slika 7-22 Izgled Windows 95 grafičkog korisničkog sučelja

Slika 7-23 Izgled Windows 3.11 grafičkog korisničkog sučelja

Slika 7-24 Struktura Windows grafičkog korisničkog sučelja

7.10.9. X Windows System

Slika 7-25 Izgled X Windows grafičkog korisničkog sučelja

Slika 7-26 Stuktura X Windows grafičkog korisničkog sučelja

8. Model korisničkog sučelja

U ovom i sljedećim poglavljima dat je prikaz rezultata vlastitog razvoja modela korisničkog sučelja primjenljivog za konstruiranje mehaničkih sklopova te opis programskih pristupa korištenih pri koncipiranju modela. Koncepcija modela korisničkog sučelja temelji se na do sada iznesenim teoretskim razmatranjima i na prikazu modela procesa konstruiranja prema [9][7].

8.1. Programski pristupi

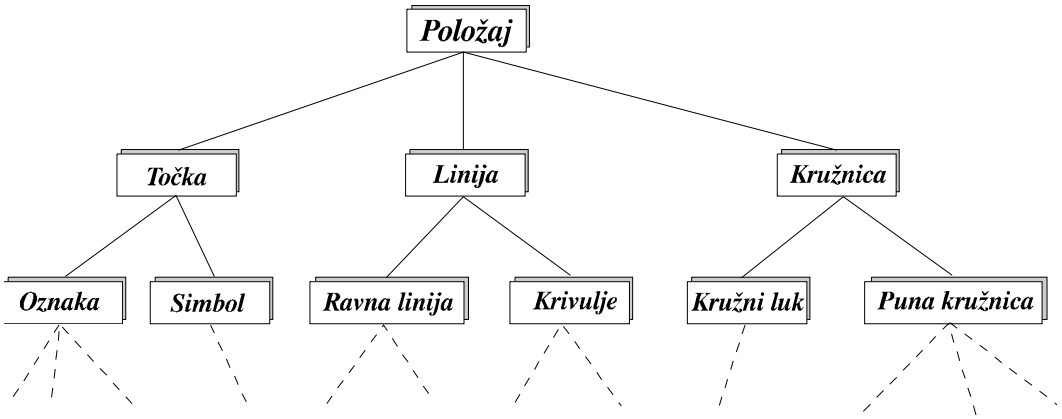
8.1.1. Objektni pristup

Prije objašnjenja objektnog pristupa mora se istaknuti da se na objektni pristup (Object Oriented Programming) ne smije gledati kao na programiranje u smislu razvoja algoritama i struktura podataka već kao način organizacije ili tehnike programiranja. Osnovi dio koji se koristi za kreiranje programa korištenjem objektnog pristupa su objekti. Objekt može biti model iz stvarnog svijeta (ljudi, stvari, životinje ...), predstavljati apstraktan kompleksan fenomen (strujanje fluida, ponori, izvori, ...) ili može predstavljati dijelove programskog sustava (stogovi, liste, grafika, ...). Funkcionalno gledajući, objekti upravljaju izvođenjem programa.

Promatrajući iz perspektive kreiranja programa tj. programiranja, najvažnija osobina objekata nije načina ponašanja kao takav već činjenica da se ponašanje objekta može opisati preko sažetog zapisa značajki ponašanja objekta u sučelje objekta. Takav

sažeti zapis ponašanja objekta dostatan je za koncipiranje modela dok se stvarno ponašanje objekta može implementirati kasnije.

Problematika i detaljan opis objektnog pristupa može se pronaći u radovima [48][37][38][49][20]. Možda jedan od najvažnijih doprinosa objektnog pristupa tehnici programiranja je uporaba nasljeđivanja svojstava objekata za određivanje relacija između (klasa) objekata (Slika 8-1). Nasljeđivanje omogućuje dodavanje funkcionalnih osobina objektima inkrementalno, te dijeljenje zajedničkih osobina, čime je omogućeno bolje koncipiranje modela.



Slika 8-1 Struktura objekata i relacije među njima

Osobine objekata se ne opisuju za svaki objekt ponaosob već se definiraju unutar klase objekata. Iz pojedine klase se deriviraju instance klase tj. objekti. Objekti koji su instance neke klase nasljeđuju osobine te klase. Derivirani objekti tvore hijerarhijsku strukturu iz koje se može vidjeti koji objekti su nasljedili koje osobine i u kakvoj su vezi sa drugim objektima.

Najvažnije osobine objektnog pristupa su:

- **nasljeđivanje**; osobine klase posjeduju svi objekti derivirani iz klase, jednostavna promijena osobina objekata promijenom u osnovnoj klasi,
- **polimorfizam**; često se miješa sa najsljeđivanjem, omogućuje različitu implementaciju osobina zajedničkog sučelja objekata,
- **modularnost (**"encapsulation"**)**; svaki objekt se može smatrati zatvorenom cjelinom, omogućuje brzo kreiranje prototipa modela, ali postoje problemi stabilnosti modela nakon implementacije,
- "**dynamic binding**"; povezivanje sa radnim bibliotekama u tijeku izvođenja programa, a ne u tijeku prevođenja i linkanja.

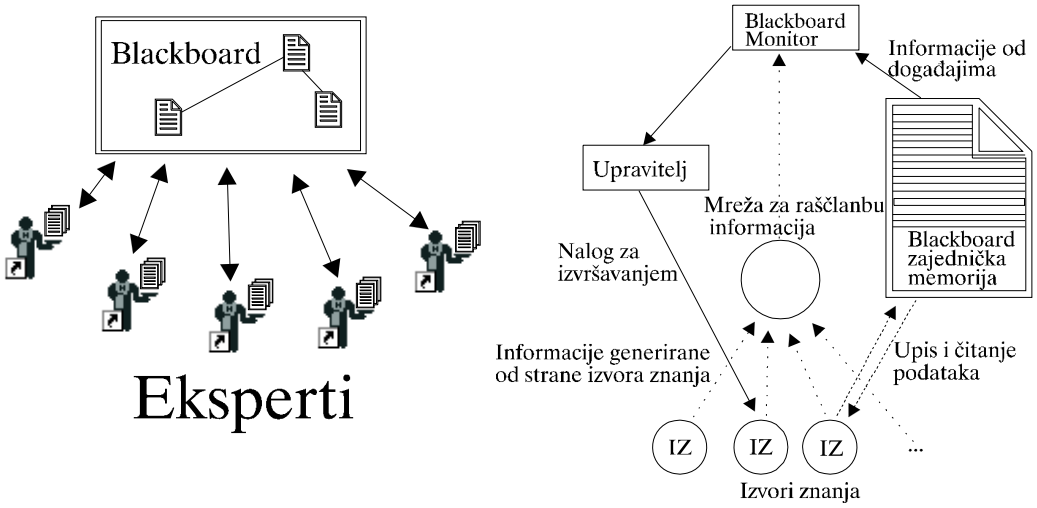
Prilikom kreiranja prototipa modela grafičkog korisničkog sučelja korišten je Visual Basic 4 razvojni alat. Koncept Visual Basic 4 razvojnog alata i programskog jezika temelji se na predefiniranim (od strane proizvođača) klasama koje programer koristi prilikom kreiranje programske aplikacije.

8.1.2. "Blackboard"

"Blackboard" model objedinjuje organizaciju informacija uključujući znanje potrebno za rješenje problema, opis načina upravljanja i način uporabe znanja u postepenom rješavanju problema. Jedna od najvažnijih osobina tog modela je stanje informacija na "blackboardu" koje u stvari čine riješenje zadanog problema. Organizacijski se "blackboard" model prema [50] sastoji od tri dijela:

- **Izvori znanja**; znanje potrebno za rješavanje problema podijeljeno je u izvore znanja, koji su međusobno neovisi i odvojeni.
- **Struktura podakata**; svi podaci (objekti) iz prostora rješenja zapisani su na jednom mjestu ("blackboard data structure"). Izvori znanja mijenjaju stanje na "blackboardu" što dovodi do približavanja rješenju problema. Komunikacije i interakcija između izvora znanja odvija se isključivo preko "blackboarda".
- **Upravljanje**; ovaj dio "blackborda" je zadužen za odlučivanje koji izvor znanja će bit upotrebljen i kada te koje mu podatke treba proslijediti.

Na slici 6-2 prikazan je konceptualni i implementacijski model "blackboard" modela.



Slika 8-2 Prikaz konceptualnog i implementacijskog modela blackboard modela

Podaci na “blackboardu” su organizirani u razine. Pri tome svaka razina na “blackboardu” odgovara jednoj razini apstrakcije podataka. Mehanizam “blackboard” modela je upravljan događajima. Događaj je bilo kakva promijena informacija na “blackboardu”. Na nekim sustavima događaj može aktivirati neki predefinirani skup izvora znanja. U složenijim sustavima nakon pojave događaja, na osnovu stanja “blackboarda” donosi se odluka o daljnjim akcijama tj. o tome koji ili koje izvore znanja treba koristiti.

Izvori znanja se mogu pisati kao procedure, ili što je češći slučaj da se kodiraju u obliku IF-THEN pravila. Opis programskih sustava koji se baziraju na “blackborad” modelu opisani su u [50][51][52][53].

8.2. Opis modela korisničkog sučelja

Iz dosadašnjeg izlaganja može se zaključiti da je korisničko sučelje korisnikov pogled u programski sustav i da je ono najeeksponiraniji dio svakog programskog sustava. Sva komunikacija koja se odvija na relaciji korisnik-računalo i računalo-korisnik prolaz u nekom obliku kroz korisničko sučelje. Zadatak korisničkog sučelja je:

- ostvarivanje komunikacije između korisnika i računala,
- uspostava komunikacije između pojedinih podsustava programskog sustava,
- kontrola tijeka i prioriteta informacija generiranih u programskom sustavu.

Informacije koje se generiraju mogu se podijeliti u dvije vrste: informacije o stanju i tijeku procesa konstruiranja te informacije generirane u pojedinim podsustavima.

Bilo da se radi o informacijama o tijeku procesa konstruiranja ili informacijama generiranim u pojednim podsustavima, postoji potreba za prikazom tih informacija. Način prikaza pojednih informacija ovisi o prirodi informacije i o odabranom načinu prikaza (od strane korisnika). Idealno bi bilo kada bi se sve informacije mogle prikazati u “svim” oblicima bez ograničenja, ali to nije slučaj, stoga je potrebno odrediti koje se informacije mogu prikazati na koje načine, te koji je “najbolji” oblik prikaza pojedinih informacija [21][54]. Način prikaza informacija ovisi u osobnim željama korisnika, tijeku procesa konstruiranja i konstrukcijskm zadatku (kontrola konstrukcije, varijacija konstrukcije, nova konstrukcija). Uz potrebu prikaza informacija korisniku, postoji i potreba za “prikazom” informacija generiranih od strane korisnika računalu tj. programskom sustavu. Korisnik može informacije zadati na razne načine, npr. preko datoteke, eksplicitno u nekom formatu, grafički (odabirom neke opcije ili nekog dijela grafičkog sučelja), redirekcijom itd. Različiti oblici zadavanja informacija moraju se obraditi tj. pretvoriti u oblik koji mogu “razumjeti” pojedini dijelovi programskog sustava. Kako po

samoj koncepciji programskog sustava [9][7][10] dijelovi sustava tj. pojedini podsustavi mogu biti i komercijalne aplikacije (aplikacije koje su proizvod različitih programerskih kuća), time kompatibilnost informacija nije nužno zajamčena. Dakle jedna od zadaća korisničkog sučelja je i pretvorba informacije iz jednog oblika zapisa u drugi (ovisno o potrebama programe aplikacije).

Kako korisnik može zahtijevati neke podatke prikazane na nekoliko različitih oblika (grafički, tekstualno, dijagramski, tablično). Da bi se podaci mogli ispravno prikazati na zaslonu računala korisničko sučelje mora imati potrebne informacije o svim podacima koje se generiraju u tijeku rada programskog sustava (bilo onih vezanih za rad sustava, bilo onih vezanih za proces konstruiranja). Potrebne inforamcije su: informacije o mogućim i podrazumijevajućim načinima prikaza, informacije o nazivu i formatu zapisa podataka te informacije o mjestu i načinu dobave tražene informacije.

Korisnik mora u svakom trenutku imati potpuni uvid u stanje tijeka procesa konstruiranja te u stanje samog programskog sustava. Jedna od funkcija korisničkog sučelja je stvaranje privida da je korisnik dio sustava te da mu sustav u svakom trenutku "izlazi" u susret te "predviđa" njegovu daljnju akciju. Kako nije strogo određena razina stručnosti korisnika programskog sustava poželjno je ostvariti mogućnost prilagodbe načina prikaza pojedinih informacija potrebama i željama korisnika.

Korisnici sustava mogu biti osobe različitih fizičkih i umnih sposobnosti te mogu pripadati različitim kulturnim i socijalnim krugovima. Razina stručnosti može varirati od početnika do eksperta. Sve ove različitosti uvjetuju potrebu za mogućnosti prilagodbe korisničkog sučelja zahtijevima korisnika. Jedan od dijelova grafičkog korisničkog sučelja koji je najviše podložan promjenama je vizualni identitet korisničkog sučelja (raspored i veličina ikona, boje, jezik komuniciranja). Rad dijela programskog sustava zaduženog za pomoć korisniku ovisi o stručnosti korisnika (korisnik početnik će se više oslanjati na pomoć od strane sustava i tražiti će više objašnjenja u tijeku rada) i ovisno o korisnikovim željama se može varirati od jednostavne do detaljne.

U tijeku rada sustava neminovna je pojava grešaka. Greške možemo podijeliti u tri skupine:

- greške vezane za tijek procesa konstruiranja (npr. promjer ležaja je premali),
- greške nastale od strane korisnika (npr. kriva opcija, krivi podatak, itd.),
- greške vezane za rad sustava (npr. nepostojenje neke datoteke).

Greške vezane za tijek procesa konstruiranja obrađuju se u modulu za objašnjavanje. Modul za objašnjavanje nije tema ovog rada, detaljnije informacije se mogu pronaći u [9][11].

Ukoliko dođe do pojave greške napravljene od strane korisnika, potrebno je zaustaviti izvođenje programskog sustava te obraditi grešku. Nakon obrade greške, korisniku se mora poslati obavijest o grešci, uzrocima koji su doveli do nastanka greške i mogućim akcijama za nastavak rada.

Greške vezane za rad sustava također moraju biti sustavno obrađene. Opis greške, razlog nastanka greške i objašnjenje mogućih akcija za nastavak rada zapisani su u datoteci grešaka.

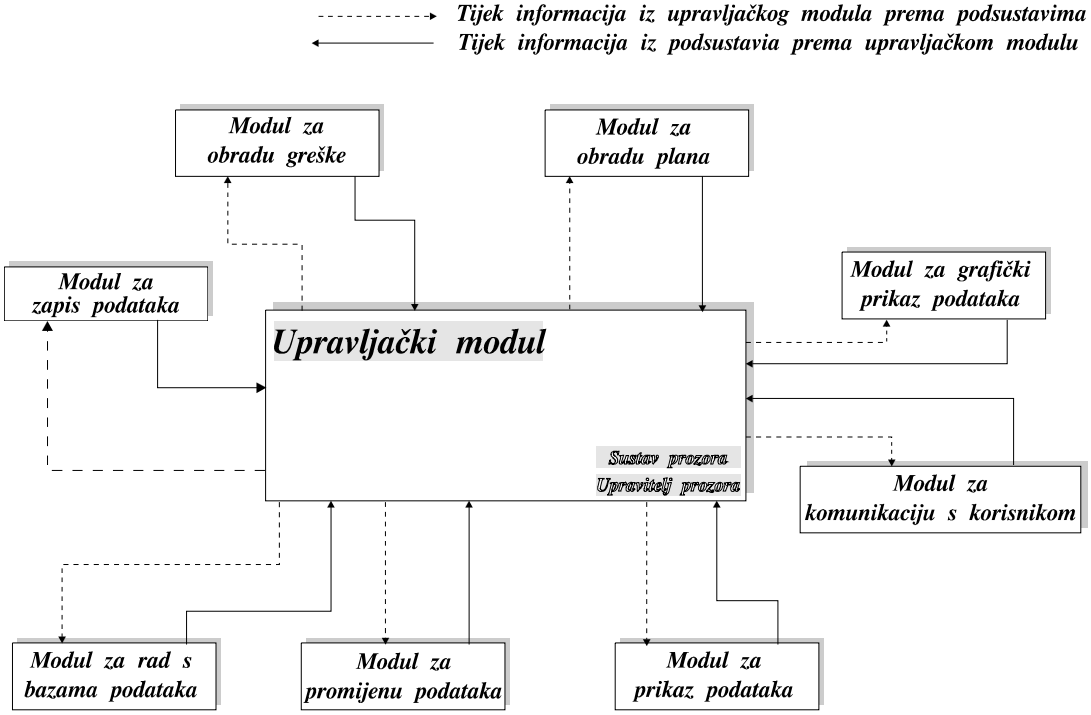
Kontrola rada sustava je ostvarena preko tri moguća načina izvodjenja programskog sustava:

- **automatski** sustav aktivira pojedini čvor i pripadnu akcijsku funkciju te na osnovu izlaznih rezultata i tablice odlučivanja [9] [7] [10] odlučuje koji će se sljedeći čvor aktivirati,
- **korak po korak** nakon izvršenja akcijske funkcije aktivnog čvora zaustavlja se izvođenje programskog sustava i čeka se naredba za nastavak rada nakon koje se aktivira čvor na osnovu izlaznih rezultata iz akcijske funkcije i tablice odluka,
- **na preskok** nakon izvršenja akcijske funkcije aktivnog čvora zaustavlja se izvođenje programskog sustava i čeka se da korisnik eksplicitno zada čvor koji će se sljedeći aktivirati, korisnikov izbor nije ograničen niti upravljan izlaznim rezultatima iz akcijske funkcije i tablicom odluka.

Vrijednosti svih podataka i informacije o podacima moraju u svakom trenutku biti dostupni pojedinim podsustavima programskog sustava ili korisniku. Na osnovu proučavanja [49] načina na koji konstruktor konstruira može se zaključiti da je mogućnost iterativnosit i podrška “šta ako” principu jedan od važnijih zadataka koje programski sustav, a i korisničko sučelja mora podržati. Iz navedenog slijedi da se mora voditi evidencija o promijeni vrijednosti podataka u tijeku procesa konstruiranja. Ovo je nužno i zbog mogućnosti poništavanja razultata posljednje operacije (“UNDO” mehanizam). Kako je proces konstruiranje u većini slučajeva dugotrajan mora se omogućiti zapis trenutnog stanja na disk te nastavak rad kasnije na mjestu gdje je proces konstruiranja prekinut.

8.2.1. Funkcionalna struktura

Na osnovu do sada navedenog slijedi funkcionalna struktura korisničkog sučelja. Osnovni dijelovi funkcionalne strukture korisničkog sučelja i tijek informacija su prikazani na slici 6-3. Sustav prozora i upravitelj prozora ovise o upravljačkom sustavu i o instaliranom grafičkom korisničkom sučelju upravljačkog sustava na kojem će model biti implementiran. Veza između modela korisničkog sučelja i instaliranog grafičkog korisničkog sučelja upravljačkog sustava ostvaruje se preko procedura zapisanih u sustavskim bibliotekama.



Slika 8-3 Funkcionalni prikaz grafičkog korisničkog sučelja

Iz slike se mogu izdvojiti osnovni moduli modela grafičkog korisničkog sučelja:

- **Upravljački modul -** ovo je glavni modul, njegova zadaća je upravljanje i kontrola rada korisničkog sučelja i tijeka informacija,
- **Modul za zapis podataka -** na modulu za zapis podataka su zapisane vrijednosti svih podatka koji se generiraju u tijeku rada programskog sustava,
- **Modul za obradu greške -** ovaj modul obađuje sve sustavske greške nastale tijekom rada programskog sustava,
- **Modul za obradu plana -** svi događaji vezani za eksploataciju plana obrađuju se u ovom modulu,

- **Modul za grafički prikaz plana -** zadatak modula je grafički prikaz strukture plana,
- **Modul za komunikaciju s korisnikom -** zadatak ovog modula je interakcija s korisnikom,
- **Modul za rad s bazama podataka -** posredstvom ovog modula se učitavaju i zapisuju podaci iz i u bazu podataka,
- **Modul za promjenu podataka -** zadatak ovog modula je ažuriranje podataka u bazi plana.
- **Modul za prikaz podataka**; zadatak modula je generiranje različitih oblika prikaza podataka.

8.2.2. Upravljački modul

Središnji dio korisničkog sučelja čini upravljački modul. Upravljački modul je u direktnoj vezi s upraviteljem prozora [48][49] tj. on je produžetak sustava prozora te je time podložan svim promjenama parametara sustava prozora i reagira na sve sustavske poruke. Uloga upravljačkog modula je:

- provjera postojanja svih datoteka programskog sustava,
- provjera zapisa u datotekama potrebnim za inicijalizaciju programskog sustava,
- postavljanje početnih vrijednosti,
- nadzor rada programskog sustava,
- nadzor i upravljanje tijekom informacija,
- ostvarivanje komunikacije između dijelova programskog sustava,
- ostvarivanje komunikacije s vanjskim programskim aplikacijama.

8.2.3. Modul za obradu plana

Zadatak ovog modula je učitavanje i grafički prikaz plana. Zapis plana na osnovu kojega se plan prikazuje dobiva se od upravitelja plana [10]. Grafički prikaz plana je u obliku strukture stabla s prikazom početnog čvora na vrhu stabla. Tijekom rada sustava tj. provođenja procesa konstruiranja po učitanom planu, prikaz pređenih čvorova se mijenja te se razlikuje od trenutno aktivnog čvora i čvorova koji nisu pređeni. Prikaz veza između pređenih čvorova se također mijenja, tako da korisnik može vizuelno vidjeti put kojim je prošao do trenutne pozicije u planu. Pređeni put se može zapisati, te ga je moguće učitati u slučaju potrebe (nastavak rada, varijacija konstrukcije).

8.2.4. Modul za zapis podataka

Modul za zapis podataka (u daljnjem teksu “blackboard”) realiziran je na principu “blackboarda” modela [50] [51] [52] [53]. Implementirani oblik “blackboarda” modela u

realizaciji modula za zapis podataka je modificiran u odnosu na teoretski opis. Na modulu za zapis podataka su zapisani svi podaci koji se generiraju u tijeku rada programskog sustava, ali zapisani podaci ne čine rješenje nego samo dio rješenja konstrukcijskog zadatka.

Modul za zapis podataka podijeljen je u dva dijela:

- dio u kojem su zapisane vrijednosti i atributi podataka, te promijena vrijednosti podataka vezanih za proces konstruiranja,
- dio u kojem su zapisane vrijednosti i atributi podataka vezanih za rad programskog sustava.

Svi atributi podataka i vrijednosti podataka na "blackboard" se zapisuju u znakovnom zapisu. To je iz tog razloga što upravitelji prozora dozvoljavaju upis i čitanje podataka samo u znakovnom zapisu, što znači da se svi podaci koji nisu zapisani u navedenom obliku moraju biti transformirati u isti tijekom čitanja, ili transformirani u stvarni zapis podataka prilikom prikaza. Blackboard je organiziran kao dvostruko povezana lista [55]. Na taj način se mogu informacije vezane za čvorove plana dinamički umetati ili izbacivati (ovisno o korisnikovim akcijama) iz "blackboarda". Podaci koju su zapisani na "blackboardu" dijelom opisuju konstrukciju i kao takovi se mogu zapisati te kasnije učitati. Na taj način korisnik može nastaviti s radom gdje je prekinuo rad ili može učitati spremljenu konstrukciju te varirati neke dijelove.

Prema [9][7][10] za svaki čvor u scenariju vezane su četiri tablice (ulazna tablica, izlazna tablica, tablica ograničenja i tablica odlučivanja). Tablice sadrže vrijednosti atributa podataka koje se razmjenjuju izneđu programskog sustava i akcijske funkcije čvora (ulazna i izlazna tablica) te vrijednosti atributa podataka koji su potrebni za odabira sljedećeg čvora (tablica odlučivanja) i kontrolu izlaznih podatka akcijske funkcije (tablica ograničenja).

U izlaznoj i ulaznoj tablici zapisane su vrijednosti sljedećih atributa:

- naziv čvora,
- naziv podatka u scenariju,
- tip podatka,
- format zapisa podatka,
- vrijednost podatka,
- kratak opis podatka.

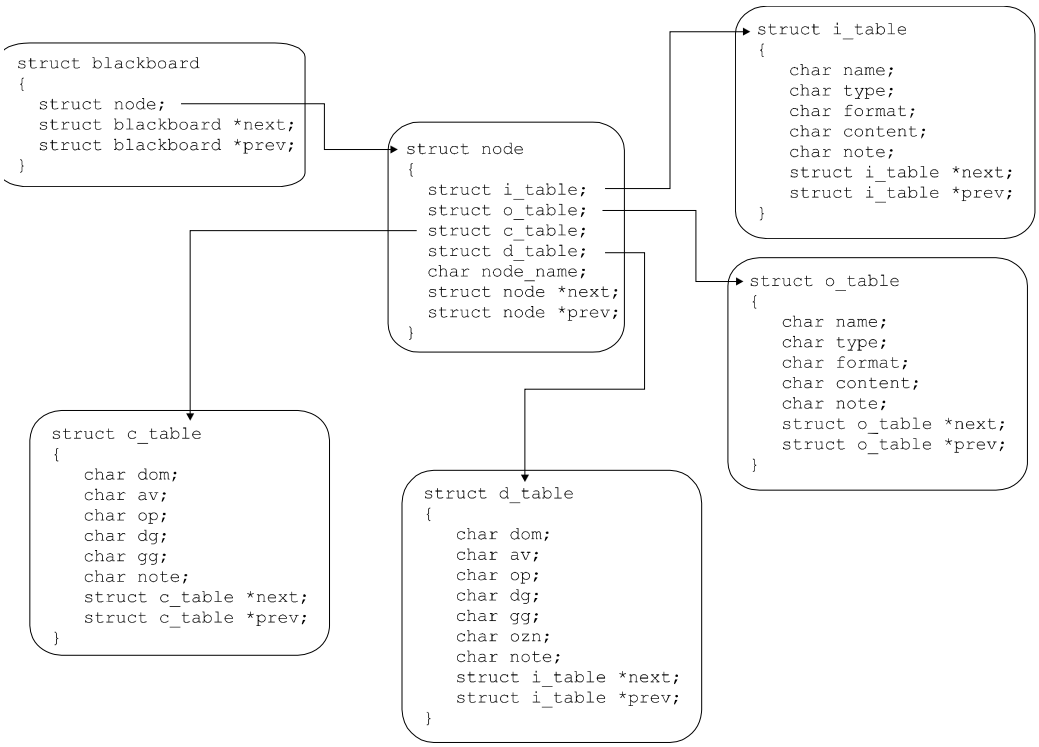
U slučaju zapisa niza podataka (polje) svaki element niza može imati vlastite vrijednosti gore navedenih atributa. Veza između elemenata niza se ostvaruje preko pokazivača na prethodni i sljedeći element u nizu.

U tablici ograničenja i tablici odlučivanja zapisane su vrijednosti sljedećih atributa:

- domena atributa,
- adresa podatka,

- operator,
- donja granica,
- gornja granica,
- naziv čvora (u tablici odlučivanja, ne postoji u tablici ograničenja),
- kratak opis.

Na slici 6-4 prikazana je struktura “blackboarda” realizirana u programskom jeziku C [56].



Slika 8-4. Struktura blackboarda u programskom jeziku C

- podaci u i_table (ulazna tablica) i o_table (izlazna tablica) strukturi:
  - node_name — naziv čvora
  - name — ime varijable
  - type — tip varijable
  - format — format zapisa vrijednosti varijable
  - content — vrijednost varijable
  - note — kratak opis varijable
- podaci u c_table (tablica ograničenja) i d_table (tablica odluka) strukturi:
  - dom — domena atributa [9][10]
  - av — adresa varijable
  - op — operator (<,<=,==,>,>=,!=, interval)
  - dg — donja granica
  - gg — gornja granica
  - ozn — oznaka čvora
  - note — kratak opis

Na “blackboardu” su zapisane samo vrijednosti atributa podataka aktiviranih čvorova tj. čvorova koji su zapisani u pređenom putu. Sadržaj “blackborda” se pretražuje preko naziva čvora. Novi čvor se dodaje na kraj liste. Tijekom dodavanja, automatski se kreiraju sve tablice vezane za čvor i upisuju se vrijednosti atributa podataka iz vanjskih datoteka. Prilikom promijene vrijednosti bilo kog podatak iz tablica sa “blackboarda” se čitaju i zapisuju podaci iz svih tablica čvora.

8.2.5. Modul za grafički prikaz podataka

Modul za grafički prikaz podataka temeljem vrijednosti atributa iz datoteke atributa podataka i vrijednosti podataka sa “blackboarda” prikazuje podatke u izabranom grafičkom obliku. Grafički se prikazuju samo nizovi podataka. Grafički prikaz vrijednosti samostalnih podataka nije dozvoljena već je moguće grafički prikazati samo promijenu vrijednosti podatka. Podržani grafički prikazi podatka su:

- tablični,
- dijagramski.

Kod tabličnog načina prikaza podataka, vrijednosti podatka i naziv podatka se upisuju u isti redak u tablici, dok se dodatne informacije o podatku prikazuju na zahtijev korisnika preko izbornika u zasebnom prozoru. U dijagramskom načinu prikaza podataka dodatne informacije o podacima se ispisuju dijelom uz prikaz podatka (vrijednost podatka), a dijelom (kratak opis podatka, format zapisa) na zahtijev korisnika, preko izbornika.

Dodatno je moguće zapisati vrijednosti nekih podataka u datoteku na disku u jednom od standardnih formata za zapis geometrijskih podataka (DXF, IGES, STEP). Takav zapis može se koristiti u nekom CAD sustavu.

8.2.6. Modul za rad sa bazama podataka

Zadatak ovog modula je učitavanje atributa podataka iz sustavskih datoteka (atributi podataka mogu biti zapisani i u tablice baze podataka). Uz učitavanje atributa podataka, ovaj modul učitava i zapis iz datoteke u kojoj su zapisane informacije o akcijskim funkcijama čvorova plana. Datoteka iz koje se učitavaju atributi podataka i datoteka iz koje se učitavaju informacije o akcijskim funkcijama imaju određenu strukturu. Strukture ovih datoteka su date u tablicama 6-1 i 6-2. Vrijednosti polja u tablicama se mogu mijenjati korištenjem programa za rad sa planovima [7] ili putem editora. Promijena vrijednosti polja iz bilo koje tablice korištenjem editora nije preporučljiva, jer su podaci povezani te bi njihova promijena narušila ispravnost plana [9]. Vrijednosti polja u tablicama koje su vezane za instalaciju programskog sustava popunjava ovaj modul

prilikom učitavanja datoteka, ali pri tome ne mijenja zapis u datotekama nego samo odgovarajuće zapise u programskom sustavu.

Tablica 8-1 Atributi podataka u sustavu

BEGIN DEF_VAR		
:PNAME:	<character>	naziv varijable u planu
:TNAME:	<character>	naziv varijable u podsustavu
:TYPE:	<character>	tip varijable
:FORMAT:	<character>	format zapisa vrijednosti varijable
:STATUS:	<character>	(FIXED,FLOAT)promjenljivost varijable
:ACCESS:	<character>	način dobave vrijednosti varijable
:SHOW:	<character>	načini prikaza vrijednosti varijable
END		

Primjer zapisa atributa podataka u datoteci:

```
BEGIN DEF_VAR
  :NAME: var_D
  :TYPE:  REAL
  :FORMAT:      10F3
  :STATUS:      FLOAT
  :ACCESS:      FILE /usr/local/demo/vratilo.dat
                KEY promjer
  :NAME: materijal
  :TYPE:  CHARCTER
  :FORMAT:      10A
  :STATUS:      FLOAT
  :ACCESS:      TERMS
END
```

Ime varijable u planu i ime varijable u programskoj aplikaciju (podsustavu) ne mora nužno biti isto tako da je potrebno navesti oba naziva varijable. Tip varijable može biti jedan od standardnih tipova podataka [57][58]: REAL, INTEGER, CHARACTER i generirani tip ARRAY, tj. polje podatka. Format zapisa podataka je sukladan načinu formatiranog ispisa ili učitavanja podataka iz programskog jezika Fortran. O statusu podatka ovisi mogućnost promijene vrijednosti nekog podatka u tijeku rada sustava. Pojedini podaci se mogu slobodno mijenjati, tada imaju status FLOAT, a ako je njihova vrijednost predodređena i ne može se mijenjati, tada imaju status FIXED. Status podataka određuje korisnik ovisno o trenutnim potrebam izvođenja programa. Inicijalno je status određen od strane kreatora plana [7]. Vrijednost polja ACCESS određuje način dobave

podatka. Podatak se može pročitati iz datoteke (FILE), sa terminala (TERMS) ili direktno iz nekog podsustava (TERMW). Ukoliko se radi o učitavanju podataka iz datoteke neophodno je zadati i ključ po kojem će podatak biti pronađen. Ključ se zadaje poslije ključne riječi KEY i može sadržavati tekst koji predhodi vrijednosti varijable ili broj redaka i stupaca (ROW broj_redka, COL broj_kolone) u datoteci na osnovu kojih će se odrediti pozicija u datoteci iz koje će se vrijednost učitati. Podatak se može prikazati na jedan ili više načina. Vrijednost polja SHOW određuje moguće oblike prikaza podatka. Podržani načini prikaza podataka su sljedeći:

- tekstualno (SHOWX),
- grafički (SHOWG),
- dijagramom (SHOWD),
- tablično (SHOWT).

Tablica 8-2 Informacije o akcijskim funkcijama u planu

BEGIN DEF_ACTION		
:ANAME:	<character>	naziv akcijske funkcije u planu
:PATH:	<character>	put do akcijske funkcije i naziv na disku
:WDIR:	<character>	radni direktorij akcijske funkcije
:STATUS:	<character>	status akcijske funkcije
:ACCESS:	<character>	način aktiviranja akcijske funkcije
END		

:ANAME: Naziv akcijske funkcije, vezan je za plan te ne mora biti isti kao naziv stvarne programske aplikacije koja se pokreće prilikom poziva akcijske funkcije.

:PATH: Put do programske aplikacije. Zapisa puta do programske aplikacije ne ovisi o upravljačkom sustavu već upravljački modul korisničkog sučelja ovisno o upravljačkom sustavu pretvara zapis iz tablice u zapis razumljiv upravljačkom sustavu. Broj znakova u imenu programske aplikacije ovisi o upravljačkom sustavu na kojem se programski sustav izvodi.

:WDIR: Radni direktorij. Za zapis radnog direktorija važe ista pravila kao i za zapis puta do programske aplikacije. Vrijednost polja iza ključne riječi određuje gdje se nalaze radne datoteke programske aplikacije akcijske funkcije.

:STATUS: Status akcijske funkcije. Status određuje na koji način programski sustav pristupa programskoj aplikaciji prilikom njenog poziva. Ukoliko je

programska aplikacija zaključana (status je određen kao APP_LOC), zaustavlja se izvođenje programa i generira se poruka o grešci. Programska aplikacija može biti zaključana u sljedećim slučajevima:

- ako se nalazi negdje na mreži i trenutno nije dostupna,
- ako se programska aplikacija ne nalazi u predodređenom putu,
- ako se programska aplikacije ne može startati iz nekog drugog razloga.

Ukoliko je status akcijske funkcije određeno kao APP_SIM, znači da se prilikom aktiviranja akcijske funkcije neće startati programska aplikacija na disku već će se njeno startanje simulirati. Ukoliko je sve u redu sa programskom aplikacijom vrijednost statusa akcijske funkcije je APP_OK. Program za kreiranje planova i korisnik mogu zadati samo vrijednost APP_OK i APP_SIM za određivanje statusa akcijske funkcije. Upis statusa APP_LOC i promijena upisanih vrijednosti obavlja upravljački modul u trenutku startanja pojedine programske aplikacije.

:ACCESS: Način aktiviranja akcijske funkcije. Podržani su sljedeći načini aktiviranja akcijske funkcije:

- NETWORK mrežna adresa. Startanje programske aplikacije preko lokalne ili globalne mreže. Iza riječi NETWORK navodi se naziv računala, sa kojeg se starta programska aplikacija, u mreži ili njegova mrežna adresa.
- LOCAL. Programska aplikacije se nalazi na lokalnom računalu.
- SYSTEM. Određuje da je programska aplikacije dio distribucije programskog sustava. Prijenos podataka između programske aplikacije i programskog sustava ostvaren je preko sustava zajedničke memorije [57] u ovom slučaju nije potrebno navoditi put do programske aplikacije i radni direktorij.

Primjer zapisa u datoteci akcijskih funkcija:

```
BEGIN DEF_ACTION
  :NAME: vratilo
  :PATH:  /usr/local/demo/vratilo.exe
  :WDIR:  /usr/local/demo/stupanj1/
  :STATUS:        APP_OK
  :ACCESS:        NETWORK 161.53.117.121
  :NAME: lezaj
  :PATH:  /usr1/lezaji/lezaj.exe
  :WDIR:  /usr1/lezaji/
  :STATUS:        APP_OK
  :ACCESS:        LOCAL
END
```

8.2.7. Modul za obradu greške

Ukoliko dođe do pojave greške u programskom sustavu, generira se sifra greške. Na osnovu sifre iz datoteke sa informacijama o sustavskim greškama modul za obradu greške učitava podatke o uzroku nastanka greške i akcijama koje korisnik može poduzeti za ispravljanje nastale greške i nastavak rada. Zatim se obavještava korisnika o nastaloj grešci i mogućim akcijama za ispravljanje greške i nastavak rada. Sve sustavske greške su sustavno obrađene i zapisane u datoteci informacija o greškama. Ova je datoteka (tablica 7-3) otvorena za promijenu i prilagođenje potrebama korisnika.

Tablica 8-3 Primjer datoteke s informacijama o grešakama

Sifra	Poruka	Objašnjenje
2000	Greška prilikom učitavanja plana	Potrebno je provjeriti, da li je izabrana datoteka ispravna datoteka plana te da li datoteka sadrži sve podatke potrebne za izvršenje plana.
2001	Greška prilikom učitavanja puta	Potrebno je provjeriti, da li je izabrana datoteka ispravna te da li sadržaj datoteke odgovara specifikaciji za zapis puta
2002	Greška prilikom učitavanja podataka o konstrukciji	Potrebno je provjeriti, da li je izabrana datoteka ispravna te da li sadržaj datoteke odgovara specifikaciji za zapis puta
...	...	

8.2.8. Modul za promijenu podataka

Zadatak modula za promijenu podataka je dodavanje, promijena i brisanje podataka na “blackboardu”. U tijeku rada programskog sustava na zahtijev upravitelja plana preko upravljačkog modula modul za promijenu podataka dodaje novi element u listi čvorova te inicijalizira vrijednosti svih tablica novog čvora. Prilikom poziva akcijske funkcije modul sa “blackboarda” učitava vrijednosti ulazne tablice, a nakon završetka rada akcijske funkcije zapisuje vrijednosti iz izlazne tablice na “blackboard”.

Promijena vrijednosti podataka ovisi o statusu podatka (FLOAT, FIXED, USERDEF). U slučaju promijene vrijednosti podatka od strane korisnika, podatak dobiva novi status (USERDEF) na osnovu kojeg modul za promijenu podataka može razlučiti između “normalnih” podataka koji se mijenjaju kroz plan i onih čija vrijednost ovisi o željama korisnika. Ovdje se mora napomenuti da promijena podataka od strane korisnika može ugroziti normalno izvođenje plana, a prilikom kontrole ili varijacije konstrukcije, podatak koji je promijenjen od strane korisnika može biti promijenjen od strane modula za promijenu podataka ukoliko je vrijednost tog podatka jedna od vrijednosti koje proizlaze iz izlazne tablice akcijske funkcije. Modul za promijenu podataka će u tom slučaju obavijestiti korisnika o predstojećoj promijeni te će zahtijevati potvrdu korisnika

za dozvolu promijene. Ukoliko korisnik ne dozvoili promijenu vrijednost podataka će ostati nepromijenjena.

8.2.9. Modul za prikaz podataka

Funkcija modula za prikaz podataka je slična funkciji modula za promijenu podataka s tom razlikom da se putem modula za prikaz podataka ne mogu mijenjati vrijednosti podataka već je moguće samo pregledati sadržaj "blackboarda". Ovaj modul na zahtijev korisnika, preko modula za komunikaciju s korisnikom, prikazuje sve vrijednosti atributa podataka iz tablica čvorova u tekstualnom zapisu.

8.2.10. Modul za komunikaciju s korisnikom

Ovaj modul tijekom rada programskog sustava ostvaruje komunikaciju s korisnikom tj. nadzire sve korisnikove akcije i aktivira prozore za unos podataka i uključivanje pojednih opcija. Oblici prozora, broj opcija i podataka koji se mogu zadati u pojedinom prozoru su predefinirani od strane korisnika u datotekama za prilagođavanje rada sustava. Svaki korisnik programskog sustava može učitati svoj određeni oblik prozora, količinu i raspored opcija i podatka za svaki od prozora programskog sustava. Za potrebe korisnika, od strane kreatora sustava date su datoteke za prilogođavanje sa predodređenim (default) oblikom prozora, količinom i rasporedom opcija i podataka.

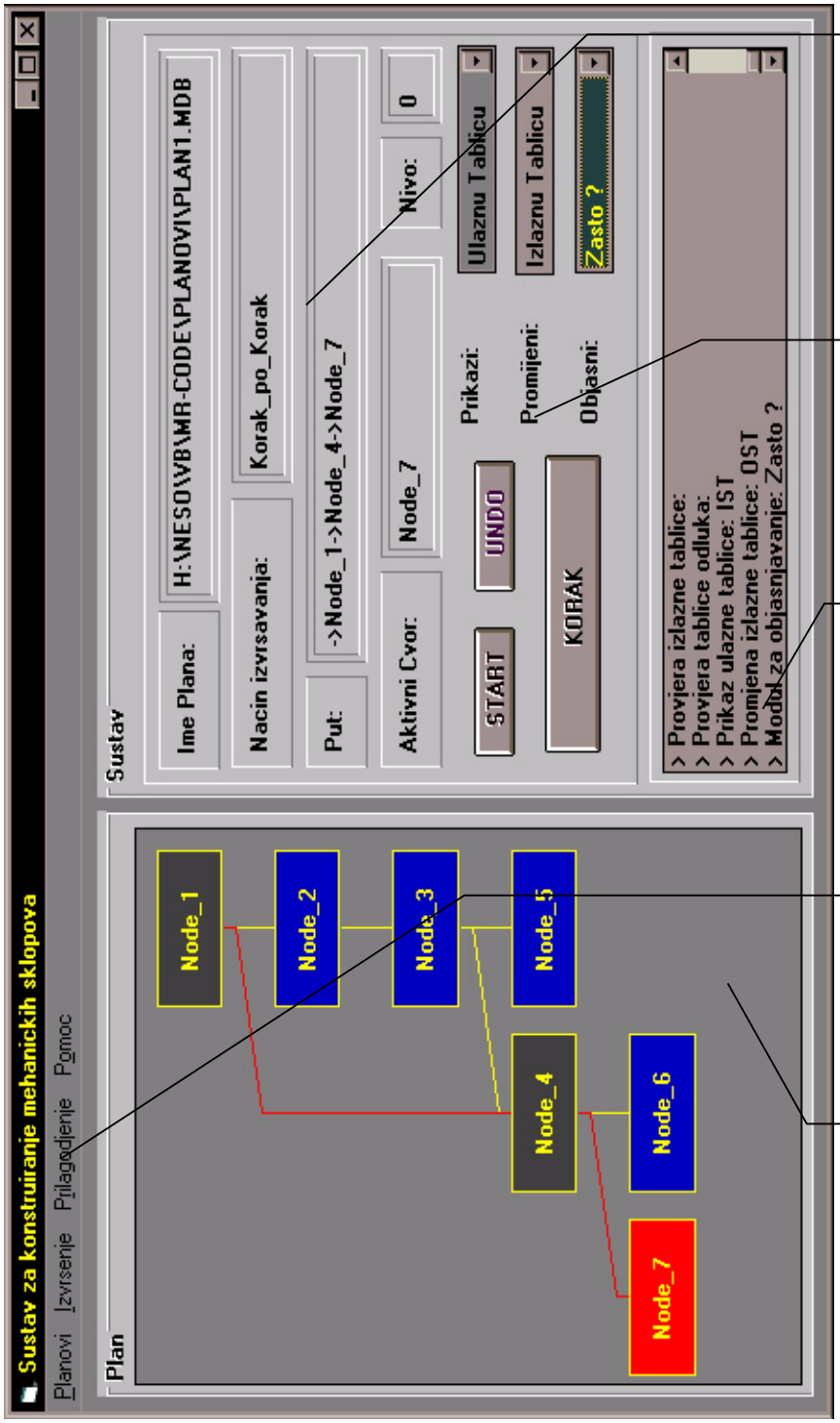
Zapis u datotekama za prilagođavanje korisničkog sučelja se mijenja preko programa za promijenu izgleda korisničkog sučelja, a nikako preko editora. Ovaj program nije sastavni dio programskog sustava, niti je dio ovog rada. Ovdje će se prikazati samo osnovne značajke programa za prilagođenje korisničkog sučelja. Rad programa za prilagođenje korisničkog sučelja mora bit neovisan o operacijskom sustavu i upravitelju prozora na kojem se program izvodi. Rezultat rada ovog programa moraju biti tekstualne datoteke. Zapis u datotekama ne smije ovisiti o sustavu prozora i upravitelju prozora.

9. Opis prototipa modela korisničkog sučelja

Prototip korisničkog sučelja usko je povezan sa bazom scenarija koja je kreirana tijekom testiranja i implementacije programa za kreiranje baze scenarija ekspertnog CAD sustava [7]. Pojedine tablice baze scenarija su nadopunjene poljima koja sadrže vrijednosi podataka. Kreirane su i dodatne tablice iz razloga implementacije blackboard mehanizma i ostvarivanja funkcije "undo" te praćenja i zapisivanja promijene vrijednosti podataka tijekom izvođenja programskog sustava tj. provođenja procesa konstruiranja.

Na slici 7-1 prikazan je izgled osnovnog prozora grafičkog korisničkog sučelja sustava za konstruiranje mehaničkih sklopova.

Osnovni prozor prototipa modela korisničkog sučelja podijeljen je u nekoliko cjelina. Prvi dio je dio za grafički prikaz baze scenarija kao strukture drveta. Čvorovi su predstavljenji pravokutnim ikonama (plava boja) s nazivom čvora u srdištu ikone. Ikone čvorova su povezane linijama koje predstavljaju veze među čvorovima, a ujedno i moguće predefinirane puteve prolaska kroz čvorove tijekom procesa konstruiranja. Prilikom provođenja procesa konstruiranja boja ikona čvorova se mijenja označujući pređene čvorove (smeđa boja) te aktivni čvor (crvena boja). Pređeni put je prikazan crvenim linijama dok su normalne veze prikazane žutom bojom. Drugi dio je dio s osnovnim informacijama o tijeku procesa konstruiranja, a to su informacije o nazivu učitanog scenarija, način izvršavanja scenarija, pređeni put, naziv aktivnog čvora i broj razina izvršavanja.



informacije o izvršavanju scenarija

kontrola izvršavanja uèitanog scena-rija

poruke programskog sustava

traka sa izbornicima

grafièki prikaz uèitanog scenarija

Slika 9-1 Osnovni prozor prototipa modela grafičkog korisničkog sučelja

Razina izvršavanja je vezana za mogućnost da akcijska funkcija bude drugi scenarij koji se mogao izvršiti. U tom slučaju se starta isti programski sustava sa predefiniranim automatskim načinom izvršavanja i scenarijem, a broj razina se poveća za jedan. Treći dio osnovnog prozora je dio za kontrolu izvršavanja scenarija te kontrolu prikaza inforamcija. Naredbe za kontrolu izvođenja su sljedeće:

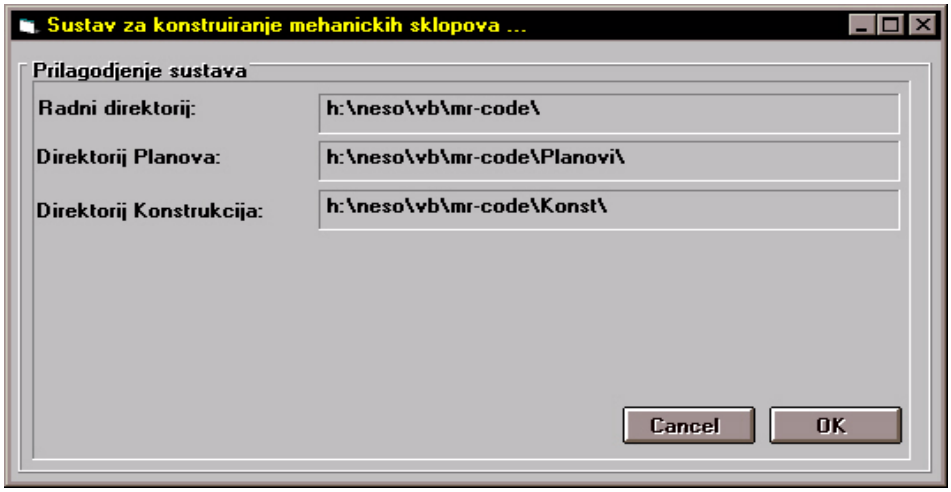
START - pritiskom na dugme s natpisom START započinje se izvršavanje učitanog scenarija,

KORAK - pritiskom na dugme s natpisom KORAK nastavlja se izvršavanje scenarija aktiviranjem sljedećeg čvora (ovisno o tablici odluka i vrijednostima u izlaznoj tablici akcijske funkcije aktivnog čvora) ili skok na sljedeći čvor u učitanom putu ukoliko se radi o nastavku rada,

UNDO - pritiskom na dugme s natpisom UNDO program se vraća na stanje koje je prethodilo trenutnom.

List kontrole u trećem dijelu osnovnog prozora služe za izbor rada s podacima. Sustav omogućuje prikaz vrijednosti i informacija o podacima, te promijenu vrijednosti podataka. U slučaju promijene vrijednosti podataka, dozvoljena je promijena samo vrijednosti podataka u ulaznoj i izlaznoj tablici aktivnog čvora. U četvrtom dijelu osnovnog prozora prikazane su poruke upravljačkog sustava i informacije o trenutnim akcijama programskog sustava.

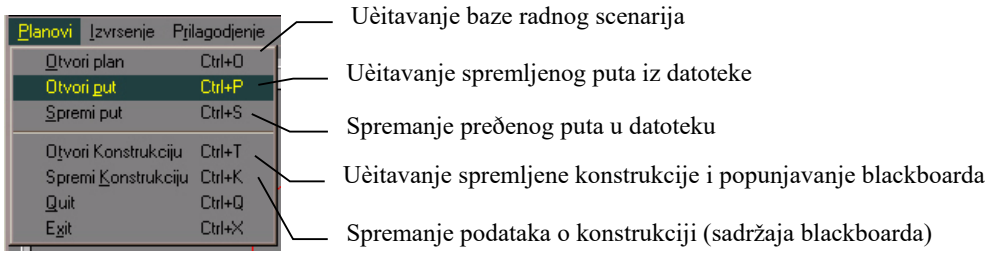
Osim opisanih dijelova osnovnog prozoru korisničkog sučelja nalazi se i traka s izbornicim preko koje se također mogu zadavati naredbe.



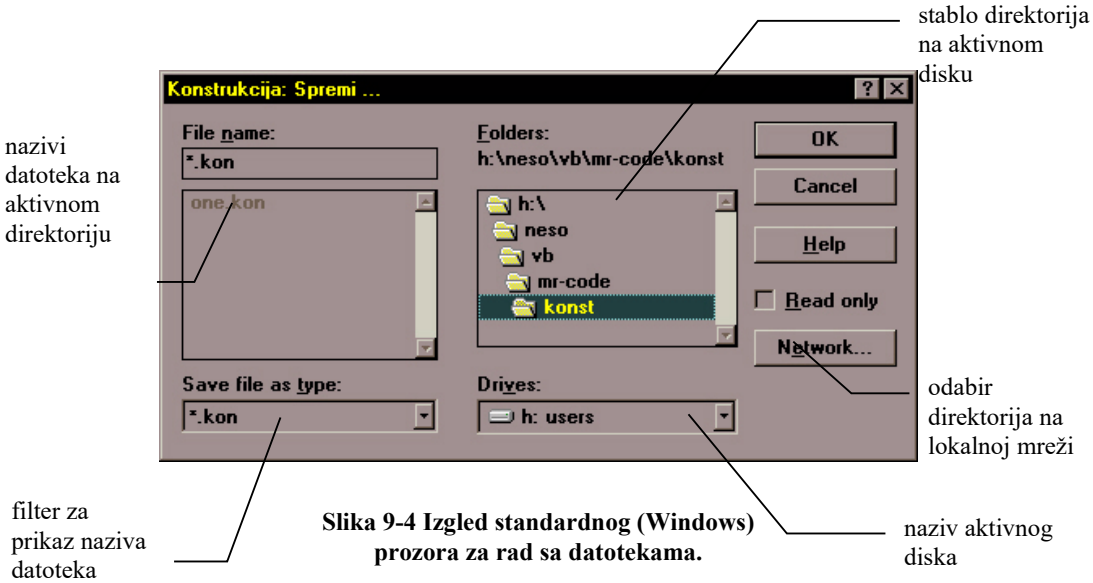
Slika 9-2 Prozor za prilagođenje radne okoline

Prije učitavanja radnog scenaraija korisnik mora prilagoditi radnu okolinu. Radna okolina se prilagođava izborom opcije ***Okolina*** padajućeg izbornika ***Prilagodjenje***. Prilikom odabira ove opcije na ekranu će se pojaviti prozor (slika 7-2.) u kojem je potrebno zapisati vrijednosti za ***Radni direktorij:***, ***Direktorij Planova:*** i ***Direktorij Konstrukcija:***. Vrijednosti upisane za ove opcije se upisuju u inicijalizacijsku datoteku *main.ini* koja se nalazi u *window* direktoriju. Sadržaj *main.ini* datoteke se učitava svaki put kada se pokrene izvođenje programskog sustava te se na osnovu zapisa iz ove datoteke postavljaju parametri neophodni za normalan rad sustava.

Nakon prilagođenja radne okoline korisnik može učitati radni plan. Učitavanje se obavlja odabirom opcije ***Otvori Plan*** sa ***Planovi*** izbornika (slika 7-3). Sve naredbe za rad sa datotekama imaju standardni izgled kao na slici 7-4.

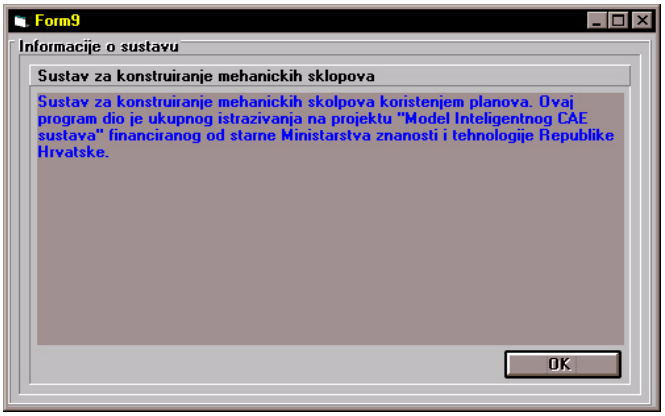


Slika 9-3 Izbornik za rad sa datotekama



Slika 9-4 Izgled standardnog (Windows) prozora za rad sa datotekama.

Izborm opcije ***Informacije*** s izbornika ***Prilagodjenje*** otvorit će se prozor s informacijama o programskom sustavu (slika 7-5).



Slika 9-5 Informacije o programskom sustavu

Nakon što je korisnik izabrao radni scenarij, programski sustav na osnovi podataka iz tablice **1_RELACIJE** (slika 7-6) aktivnog scenarija crta stablo koje grafički opisuje scenarij i veze između čvorova scenarija. Na slici 7-7 prikazan je grafički prikaz stabla scenarija u programu MS Access, a na slici 7-8 prikaz istog stabla scenarija u programskom sustavu.

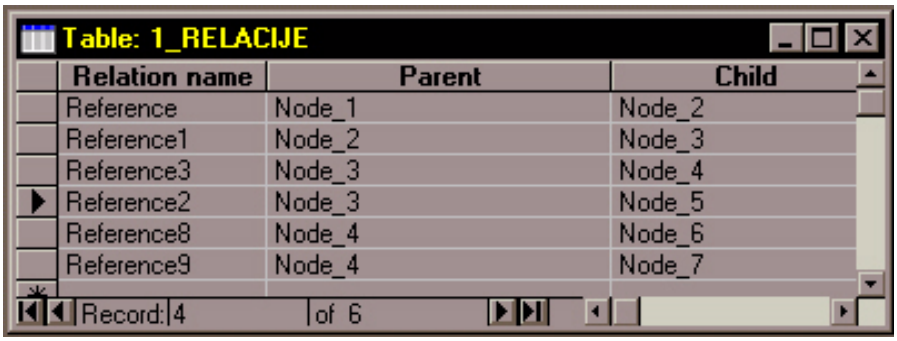
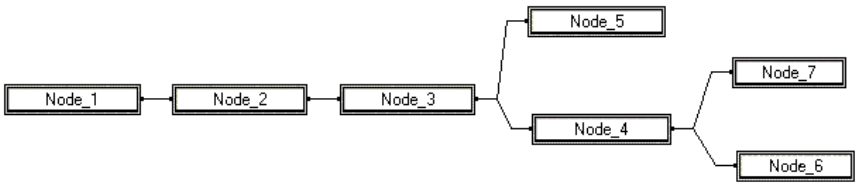


Table: 1_RELACIJE

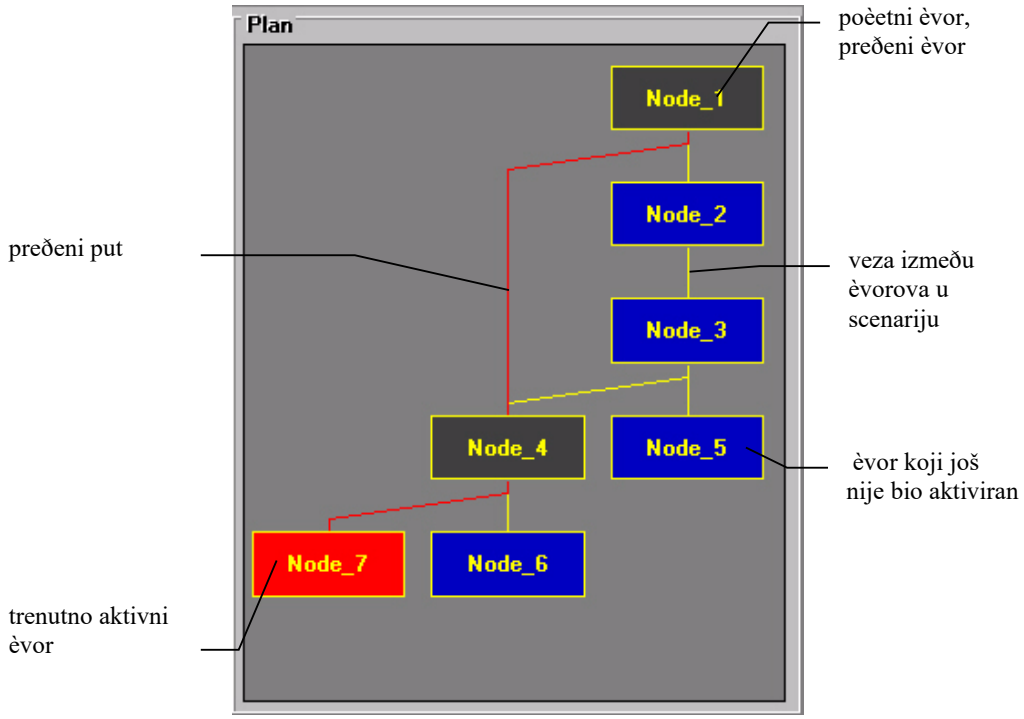
Relation name	Parent	Child
Reference	Node_1	Node_2
Reference1	Node_2	Node_3
Reference3	Node_3	Node_4
Reference2	Node_3	Node_5
Reference8	Node_4	Node_6
Reference9	Node_4	Node_7

Record: 4 of 6

Slika 9-6 Tablica realcija aktivnog scenarija

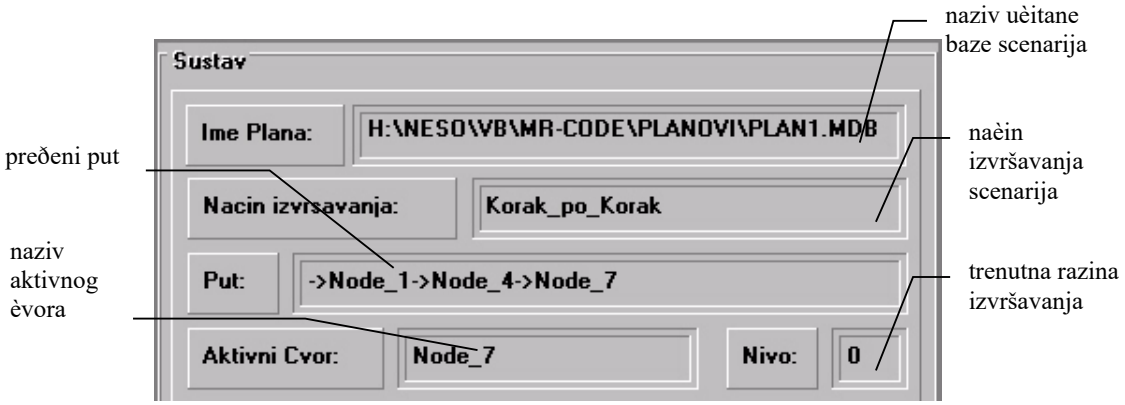


Slika 9-7 Stablo scenarija u programu MS Access



Slika 9-8 Stablo scenarija u programskom sustavu

Na slici 7-9 prikazan je dio prozora sa informacijama o radu programskog sustava i trenutnom stadiju u procesu konstruiranja te pređenim putem.



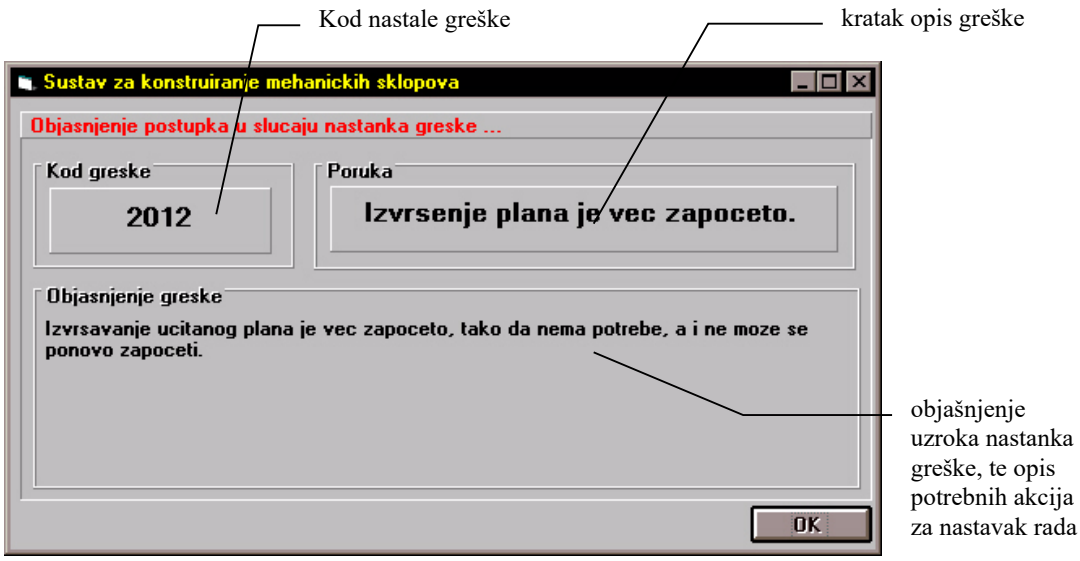
Slika 9-9 Trenutno stanje procesa konstruiranja i programskog sustava

Tokom čitavog tijeka izvršavanja scenarija korisnik može pratiti rad programskog sustava i tijek procesa konstruiranja preko sustava poruka koje se zapisuju u dijelu za ispis sustavnih poruka slika 7-10.

Slika 9-10 Kontrola za ispis poruka programskog sustava

Nakon učitavanja scenarija, korisnik može započeti izvršavanje scenarija pritiskom na tipku **START**. Ukoliko je korisnik spremio u nekom od prethodnih izvođenja pređeni put ili konstrukciju, može ih učitati te nastaviti s radom gdje je stao ili kreirati novu konstrukciju po učitanom putu. Učitana konstrukcija i pređeni put vezani su za bazu scenarija po kojoj se proces konstruiranja odvijao u trenutku spremanja konstrukcije i plana, tako da učitavanje konstrukcije ili pređenog plana koji su napravljeni na osnvu neke druge baze scenarija će dovesti do neispravnog rada programskog sustava i do nepredviđenih ponašanja u tijeku procesa konstruiranja. Veza između baze scenarija, konstrukcije i pređenog puta na sadašnjoj razini razvoja programskog sustava i korisničkog sučelja je ostvarena preko imena datoteka, a na korisniku je da učita ispravne datoteke u toku rada.

Ukoliko bi u programskom sustavu došlo do pojave greške vezana za rad programskog sustava korisnik dobije obavijest o nastaloj grešci i sve informacije vezane za nastanak greške i potrebne akcije za nastavak rada. Primjer prozora s informacijam o nastaloj grešci je prikazan na slici 7-11.



Slika 9-11 Primjer prozora s informacijam o nastaloj grešci

Kodovi svih grešaka te kratak opis i objašnjenje akcija za nastavak rada zapisani su u tablici *Code_table* u *main.mdb* bazi. Primjer zapisa informacija o sustavnim greškama dat je na slici 7-12.

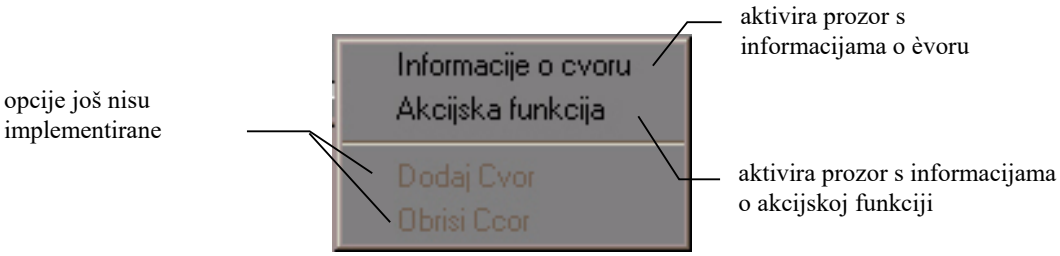
Table: Code_table

Code	Poruka	Objasnjenje
2000	Greska prilikom ucitavanja plana	Potrebno je provjeriti, da li je izabrana datoteka ispravna datoteka plana te da li datoteka sadrzi sve poı
2001	Greska prilikom ucitavanja puta	Potrebno je provjeriti, da li je izabrana datoteka ispravna te da li sadrzaj datoteke odgovara specifikaciji
2002	Greska prilikom ucitavanja podataka o konstrukciji	Potrebno je provjeriti ispravnost odabrane datoteke te njen sadrzaj.
2003	Greska prilikom postavljanja aktivnog cvora	Cvor s izabranim imenom ne postoji u planu. Potrebno je provjeriti nazive cvorova u ucitanom putu.
2004	Greska prilikom provjere izlazne datoteke	Izlazna datoteka ne postoji ili je naziv izlazne datoteke krivu naveden u planu. Potrebno je provjeriti naz
2005	Greska prilikom poziva akcijske funkcije	Akcijska funkcija zadanog naziva ne postoji u planu ili na disku. Potrebno je provjeriti ispravnost naziva
2006	Greska u radu akcijeske funkcije	Greska se pojavila u tijeku izvodjenja akcijske funkcije. Nastanak greske je moguc u slucaju djeljenja s
2008	Greska prilikom prikaza tablice	Greska je nastala prilikom ucitavanja podataka iz odabrane tablice i prikaza istih na zaslonu. Potrebno j
2009	Greska prilikom promjene tablice	Greska je nastala prilikom ucitavanja podataka iz odabrane tablice. Potrebno je provjeriti da li tablica za
2010	Plan nije ucitan	Prije pocetka rada neophodno je ucitati plan po kojem ce se raditi.
2011	Nije postavljen aktivni cvor	Da bi se zapoceo rad sa ucitanim planom, potrebno je odrediti cvor od kojega ce poceti izvodjenje plan
2007	Greska je nastala prilikom provjere izlazne tablice	Potrebno je provjeriti naziv i postojanja izlazne tablice kako u planu tako i na disku.
2012	Izvrsenje plana je vec zapoceto.	Izvrsavanje ucitanog plana je vec zapoceto, tako da nema potrebe, a i ne moze se ponovo zapoceti.
2013	Put nije ucitan	Ukoliko se zeli plan izvrsavati na osnovu nekog unaprijed pripremljenog puta tada je potrebno taj put uc
2014	Izvravanje preko ucitanog puta	Ucitan je predhodno spremljen ili pripremljen put izvrsenja te se stoga mora po njemu izvrsavati plan. Za
2015	Izvrsavanje preko ucitanog puta	Slijedeci cvor se moze izabrati putem misa samo ukoliko je zavrseno izvodjenje svih cvorova iz ucitano

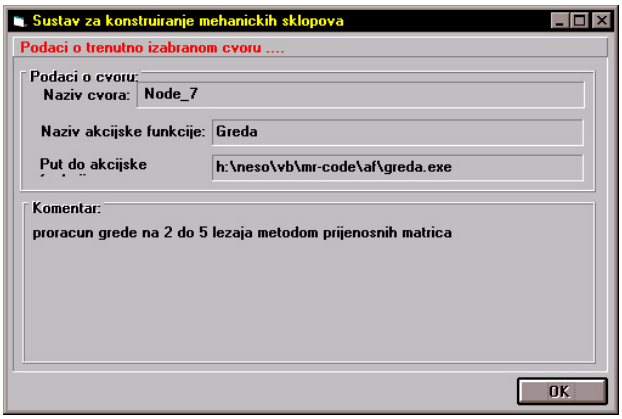
Record: 40 of 58

Slika 9-12 Primjer zapisa informacija o sustavnim greškama

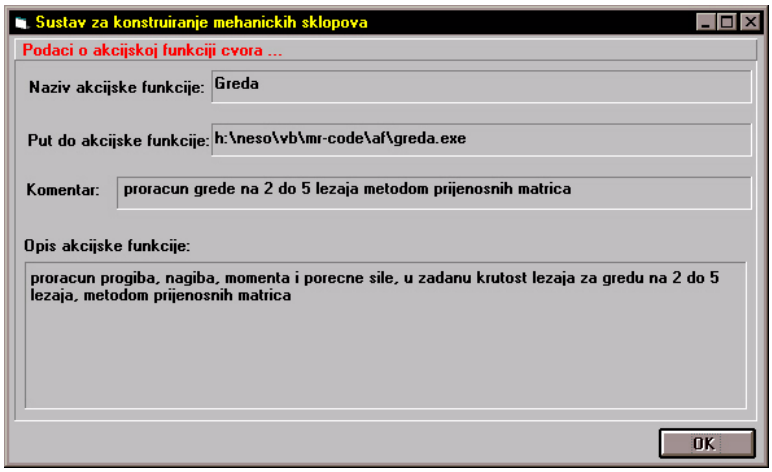
U dijelu osnovnog prozora korisničkog sučelja gdje je prikazano stablo scenarija pritiskom na desno dugme na mišu na nekoj od ikona pojavit će se izbornik (slika 7-13) na kojem se može izabrati prikaz dodatnih informacija o izabranom čvoru (slika 7-14) te o pripadajućoj akcijskoj funkciji (slika 7-15).



Slika 9-13 Iskakajući izbornik

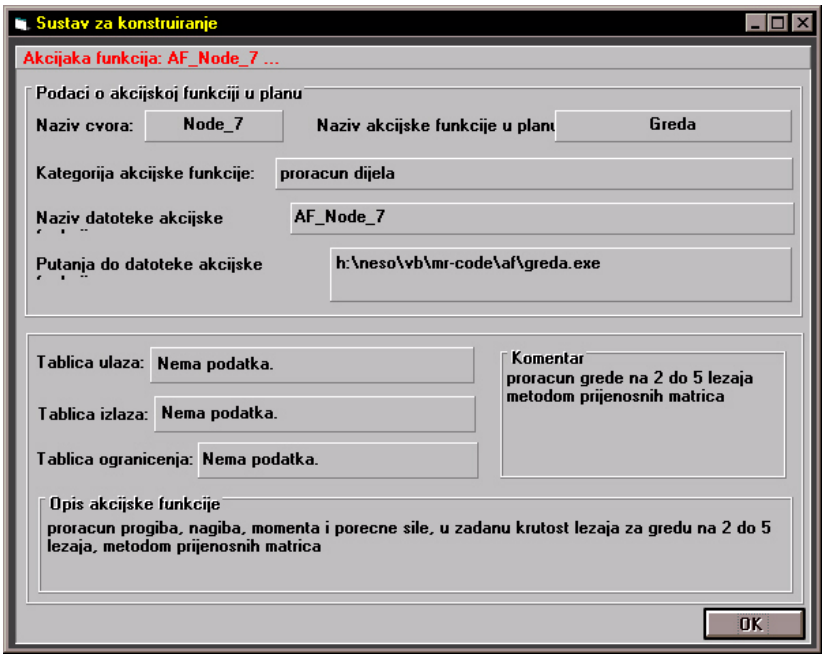


Slika 9-14 Informacije o izabranom čvoru



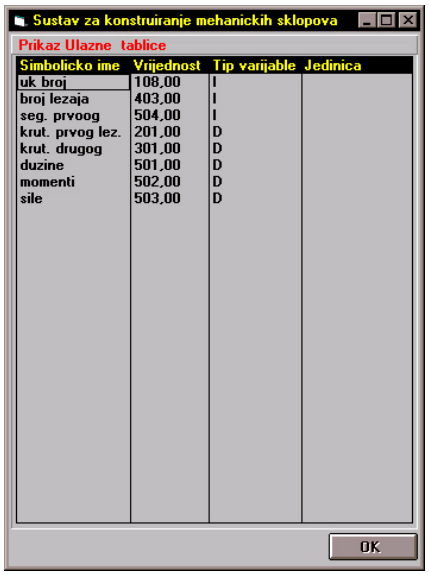
Slika 9-15 Informacije o akcijskoj funkciji izabranog čvora

Nakon pokretanja izvršavanja scenarija sustav provjerava postojanje akcijske funkcije na putu koji je zapisan u tablici *AKCIJSKE_FUNKCIJE* u bazi radnog scenarija, ukoliko akcijska funkcija tj. pripadajuća aplikacija ne postoji na zadanom putu, korisnik dobije poruku o grešci. Ukoliko se akcijska funkcija tj. pripadajuća aplikacija nalazi na zadanom putu tad se priprema ulazna tablica i starta akcijska funkcija. Prije startanja akcijske funkcije korisnik u prozoru (slika 7-16) dobija sve informacije relevantne za startanje akcijske funkcije. Zatim se pokrene akcijska funkcija.



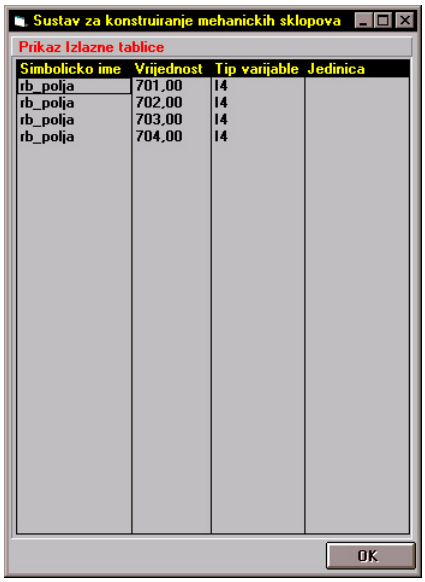
Slika 9-16 Prozor za startanje akcijske funkcije

Nakon završetka akcijske funkcije popunjava se izlazna tablica koja se ispituje da li zadovoljava kriterije zadane u tablici ograničenja. Informacije zapisane u ulaznoj, izlaznoj tablici, tablici ograničenja i tablici odluka mogu se pregledati izborom opcije za pregledavanje određene tablice u list kontroli ***Prikazi:*** na osnovnom prozoru korisničkog sučelja. Na slikama 7-17 (Ulazna tablica), 7-18 (Izlazna tablica), 7-19 (Tablica ograničenja) i 7-20 (Tablica odluka) dati su primjeri informacija zapisanim u određenoj tablici.



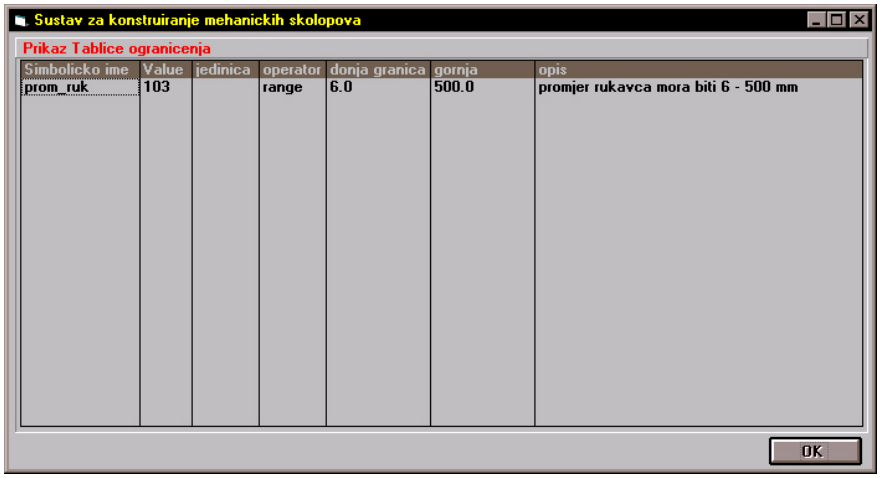
Simbolicko ime	Vrijednost	Tip varijable	Jedinica
uk broj	108,00	I	
broj lezaja	403,00	I	
seg. prvoog	504,00	I	
krut. prvog lez.	201,00	D	
krut. drugog	301,00	D	
duzine	501,00	D	
momenti	502,00	D	
sile	503,00	D	

Slika 9-17 Ulazna tablica



Simbolicko ime	Vrijednost	Tip varijable	Jedinica
rb_polja	701,00	I4	
rb_polja	702,00	I4	
rb_polja	703,00	I4	
rb_polja	704,00	I4	

Slika 9-18 Izlazna tablica



Simbolicko ime	Value	jedinica	operator	donja granica	gornja	opis
prom_ruk	103		range	6.0	500.0	promjer rukavca mora biti 6 - 500 mm

Slika 9-19 Tablica ograničenja

Sustav za konstruiranje mehanickih sklopova

Prikaz Tablice odluka

Simbolicko	Vrijednost	Jedinica	Tip varijable	Operator	Donja	Gornja	Slijedeci	Opis
VOPT	107,00		C	==	UDARNO	0	Node_4	proracun pera
VOPT	107,00		C	==	ISTOSMJERN	0	Node_5	proracun klina
T	301,00		C	==	PROMJENJIV	0	Node_5	proracun pera

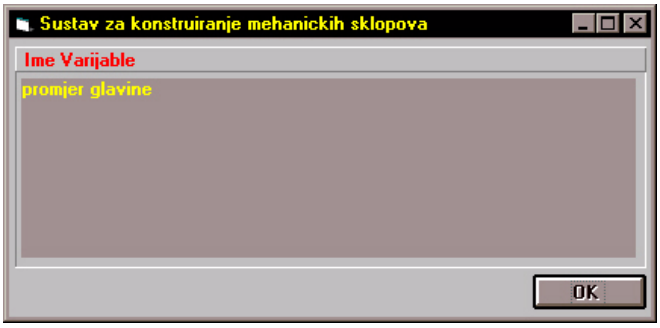
OK

Slika 9-20 Tablica odluka

Odabir oblika prikaza informacija u ulaznoj i izlaznoj tablici ostvaruje se na iskakajućem izborniku (slika 7-21) koji se pojavi nakon pritiska na desno dugme miša na podatak koji se želi prikazati grafički.

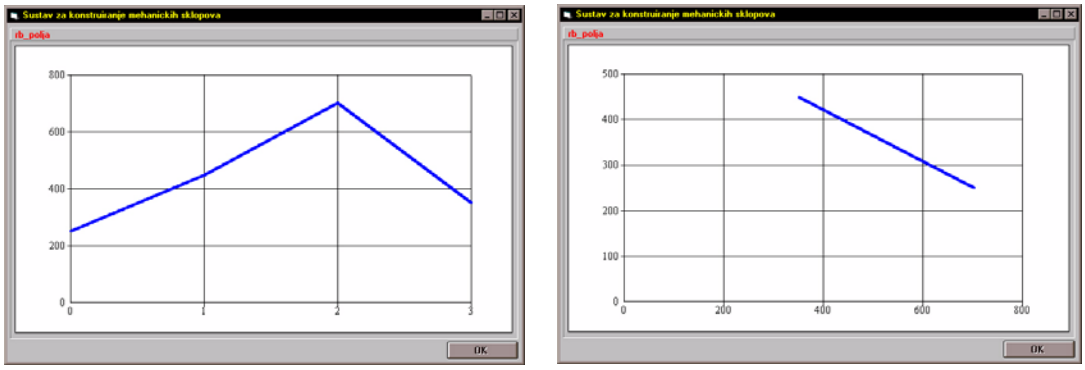
Slika 9-21 Iskakajući izbornik prikaza podataka

Na slici 7-22 prikazan je izgled prozora s informacijama o izabranim podacima.



Slika 9-22 Prozor s kratkim opisom podatka

Izborom opcije ***Prikazi Graficki [x]*** otvorit će se prozor kao na slici 7-23a na kojem su grafički prikazane vrijednosti polja izabranog na ulaznoj tablici. Izborom opcije ***Prikazi Graficki [x/y]*** otvorit će se prozor kao na slici 7-23b na kojem su također grafički prikazane vrijednosti polja izabranog na ulaznoj tablici s tom razlikom da su vrijednosti na x koordinatnoj osi vrijednosti polja do polovice dok su vrijednosti na y koordinatnoj osi vrijednosti polja iznad polovice.

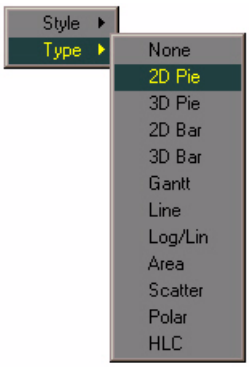


(a) **(b)**

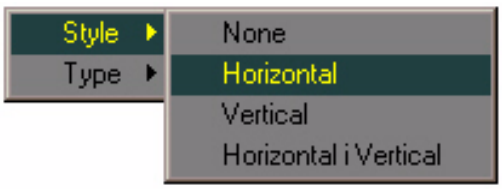
Slika 9-23 grafički prikaz vrijednosti polja

Izborom opcije ***Prikazi promjenu*** izgled prozora prikaza će biti identičan izgledu prozora na slici 8-23. Ova opcija ima smisla samo onda ukoliko se vrijednost jednog podatka mijenjala tijekom procesa konstruiranja, pa se na ovaj način može grafički vidjeti njena promjena. Različiti načini grafičkog prikaza promjene vrijednosti podatka izabiru se na isti način kao i pri grafičkom prikazu polja.

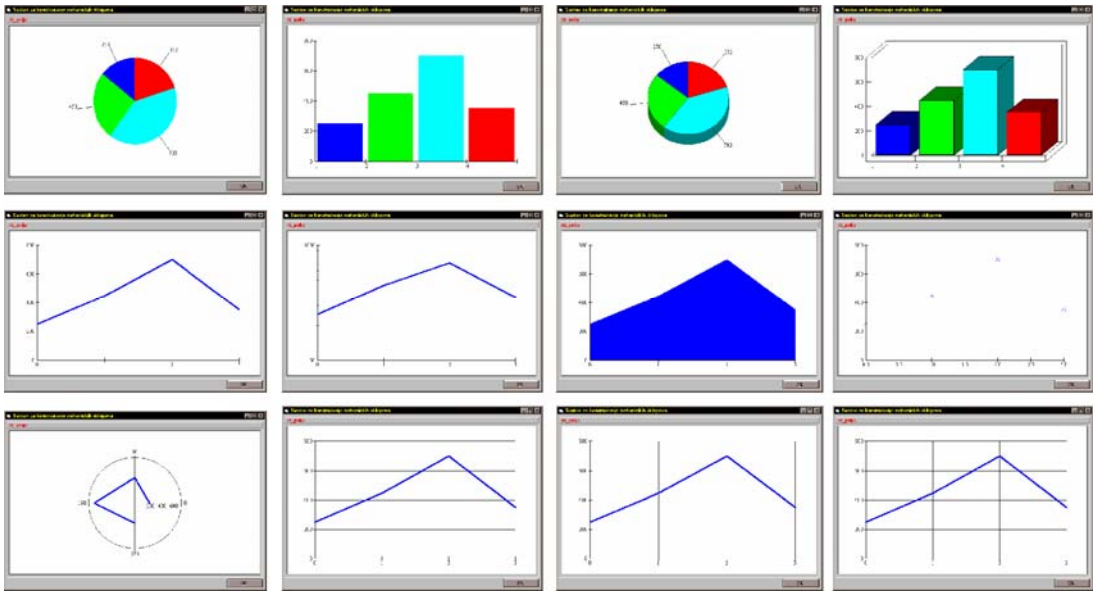
Na slikama 7-24 i 7-25 prikazane su mogućnosti izbora oblika i stila grafičkog prikaza. Primjeri prikaza ovisno o odabranom obliku i stilu prikazan je na slici 7-26.



Slika 9-24 Izbor oblika grafičkog prikaza

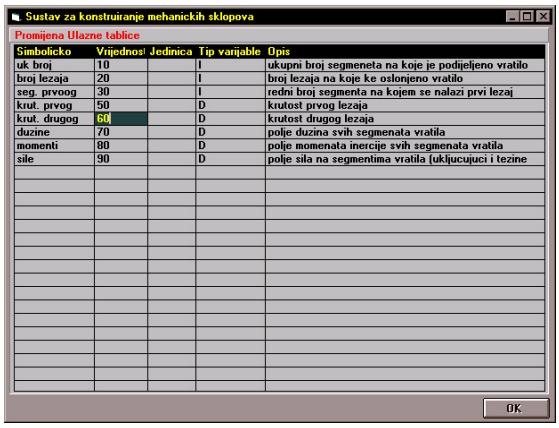


Slika 9-25 Izbor stila grafičkog prikaza



Slika 9-26 Različiti grafički prikazi podataka

Slike 7-27 (Ulazna tablica) i 7-28 (Izlazna tablica) prikazuju izgled prozora za promjenu vrijednosti i atributa podataka. Prikazani prozori se otvore nakon izbora jedne od tablica iz list kontrole za promjenu podataka ***Promijeni:***.



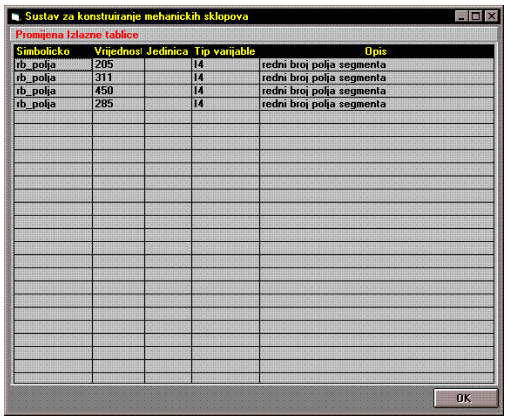
Sustav za konstruiranje mehanickih sklopova

Promijena Ulazne tablice

Simbolicko	Vrijednost	Jedinica	Tip varijable	Opis
uk broj	10		I	ukupni broj segmeneta na koje je podijeljeno vratilo
broj lezaja	20		I	broj lezaja na koje ke oslonjeno vratilo
seg. prvoog	30		I	redni broj segmenta na kojem se nalazi prvi lezaj
krut. prvog	50		D	krutost prvog lezaja
krut. drugog	60		D	krutost drugog lezaja
duzine	70		D	polje duzina svih segmenata vratila
momenti	80		D	polje momenata inercije svih segmenata vratila
sile	90		D	polje sila na segmentima vratila (ukljucujuci i tezine

OK

Slika 9-27 Prikaz prozora za promjenu ulazne tablice



Sustav za konstruiranje mehanickih sklopova

Promijena Izlazne tablice

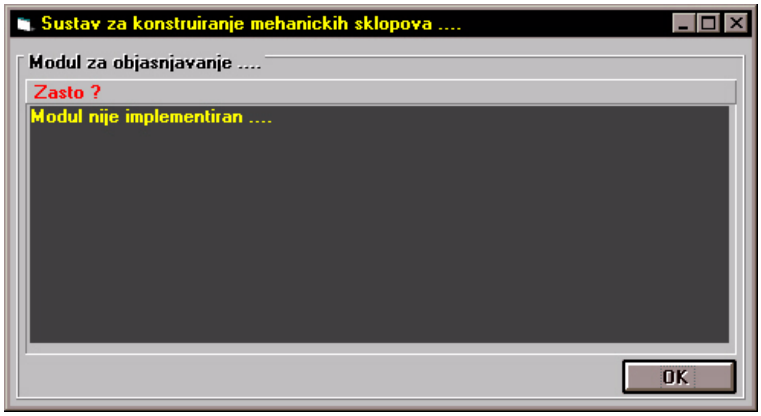
Simbolicko	Vrijednost	Jedinica	Tip varijable	Opis
rb_polja	205		I4	redni broj polja segmenta
rb_polja	311		I4	redni broj polja segmenta
rb_polja	450		I4	redni broj polja segmenta
rb_polja	285		I4	redni broj polja segmenta

OK

Slika 9-28 Prikaz prozora za promjenu izlazne tablice

U bilo kom trenutku tijekom rada programskog sustava tj. tijekom procesa konstruiranja korisnik može zatražiti pomoć i objašnjenje kako i zašto je programski sustav izabrao neki čvor ili došao do neke odluke. Prozor za poziv modula za objašnjavanje prikazan je na slici 7-29. Ovaj modul u trenutku izrade grafičkog

korisničkog sučelja i programskog sustava nije bio prilagođen za komunikaciju i prijenos argumenata ka i od programskog sustava, tako da modul nije implementiran.



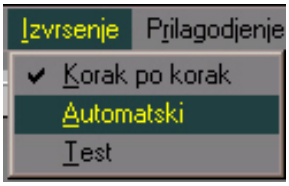
Slika 9-29 Prozor modula za objašnjavanje.

Korisnik može na početku rada ili tijekom rada programskog sustava mijenjati način izvođenja programskog sustava. Na rasplaganju su tri načina izvođenja (slika 7-30):

Automatski programski sustav na osnovu podataka iz izlazne tablice akcijake funkcije i tablice odluka automatski odlučuje koji će čvor sljedeči biti aktiviran

Korak po korak programski sustav nakon povratka iz akcijske funkcije te kontrole tablice ograničenja i tablice odluke trenutno zaustavi izvođenje programa te ostavalja mogućnos korisniku da odabirom dugmeta ***KORAK*** aktivira sljedeči čvor na osnovu tablice odluka ili može izabrati bilo koji čvor kao sljedeči

Test ova se opcija koristi prilikom testiranja rada programskog sustava i nema značajnu funkciju u tijeku eksplatacije programaskog sustava



Slika 9-30 Prikaz opcija izbornika za određivanje načina izvršavanja programskog sustava

10.Zaključak

U ovom radu su izneseni rezultati istraživanja provedenih u razvoju modela korisničkog sučelja sustava za konstruiranje mehaničkih sklopova. U uvodnim razmatranjima opisani su problemi i nedostaci današnjih korisničkih sučelja CAD programskh sustava. Također su naznačeni zadaci korisničkog sučelja, te smjernice u koncipiranju i razradi korisničkog sučelja.

Potom su prikazani različiti pristupi procesu konstruiranja, te opisane faze procesa konstruiranja i metode prikaza konstrukcijskog procesa. U okviru razmatranja procesa konstruiranja analiziran je oblik i količina informacija u tijeku procesa konstruiranja, te su naznačene značajke koje informacija mora posjedovati da bi bila uporabljiva. Prema jednom od postavljenih zadataka ovog rada model korisničkog sučelja treba omogućiti prikaz informacija i stanja procesa konstruiranja definiranih scenarijem. Iz tog razloga razmatran je prikaz procesa konstruiranja planom ("bazom scenarija"), te je prikazana struktura ekspertnog CAE sustava. Na osnovu prikaza strukture ekspertnog CAE sustava određeno je mjesto i uloga korisničkog sučelja u CAE sustavu. Iz opisa rada CAE sustava određen je tijek i količina informacija koje se generiraju u tijeku rada programskog sustava, te oblik prikaza pojedinih podataka.

Iz razmatranja opisanih u 1. i 3. glavi određeni su sljedeći zadaci korisničkog sučelja:

- ostvarivanje komunikacije s korisnikom,
- ostvarivanje komunikacije između dijelova programskog sustava,

- kontrola tijeka procesa konstruiranja,
- upravljenje radom programskog sustava,
- kontrola i prikaz informacija vezanih za odvijanje procesa konstruiranja,
- kontrola i prikaz informacija vezanih za rad programskog sustava,
- osiguranje interakcije s klasičnim programskim sustavima,
- transformacija zapisa i prijenos informacija.

Posebna je pažnja posvećena analizi postojećih grafičkih korisničkih sučelja i grafičkih standarda. Na osnovu provedene analize određen je način komunikacije između korisničkog sučelja i postojećih grafičkih korisničkih sučelja upravljačkih sustava. Zatim su analizirani pristupi u koncipiranju i razvoju korisničkih sučelja. Pri tome su opisani iskustveni, spoznajni i antropološki pristup, te pristup predviđanjem. Također su opisani modeli za formaliziranje dijaloga između čovjeka i računala (jezični i izvedbeni). Dana je poodijela korisničkih sučelja na osnovu funkcija koje nude u dvije kategorije: zadaćno orijentirana i sučelja s direktnom manipulacijom. Prilikom koncipiranja modela korisničkog sučelja direktna manipulacije je primjenjena u radu s prikazom plana dok je u radu s podacima primjenjen princip zadačno orijentiranih korisničkih sučelja.

U realizaciji modela korišteni su objektni pristup i princip "blackboarda" koji su opisani u 1. poglavlju u 6. glavi. Potom je opisan model grafičkog korisničkog sučelja. U okviru opisa prikazana je funkcionalna struktura i tijek informacija, te je naznačena uloga pojedinih dijelova modela. Potom je dan detaljn opis pojedinih dijelovi modela grafičkog korisničkog sučelja, te opisa pomoćnih sustavskih datoteka i datoteke neophodne za rad dijela modela zaduženog za pomoć u slučaju nastanke greške. U okviru opisa dijelova modela opisan je i način komuniciranja s bazom scenarija i izmjene na tablicama baze neophodnih iz razloga implementacije i testiranja prototipa modela. Na osnovu prikaza funkcionalne strukture i opisa dijelova realiziran je prototip grafičkog korisničkog sučelja u programskom jeziku Visual Basic. Realizirani prototip je povezan sa bazom scenarija preko alata za komunikaciju s programskom aplikacijom Microsoft Access. Rad akcijskih funkcija tj. programskih aplikacija je simuliran preko dodatnih tablica u bazi scenarija. Sve tablice koje čine bazu scenarija programskog sustava su modificirane tako da uz osnovne podatke sadrže i vrijednosti podataka generirane i tijeku rada programskog sustava.

Zaključeno je da se razvijeni model korisničkog sučelja dobro uklapa u model CAE sustava, ali da način komuniciranja između dijelova programskog sustava ipak neće biti dovoljo efikasan. Iz tog razloga u daljnjim istraživanjima mora se proučiti način komunikacije između modula programskog sustava te realizirati jezik i protokol komunikacije. Pristup zapisu vrijednosti podataka i atributa podataka putem "blackboard" modela pokazao se ispravnim, iz tog razloga što je omogućeno da svaki podatak,

neovisno da li je varijabla ili element polja, može imati vlastiti format zapisa, te vlastiti opis koji je neovisan o ostalim elementima. Direktno povezivanje korisnčkog sučelja tj. programskog sustava s bazom scenarija pokazalo se ispravnim, te. bi se daljni razvoj korisničkog sučelja kao i cijelog programskog sustava morao kretati u tom smjeru.

U tijeku testiranja prototipa posebna pažnja je posvećena radu dijelova prototipa zaduženih za prikaz plana i pređenog puta te dijelova zaduženih za rad UNDO mehanizma. Rad UNDO mehanizma se pokazao efikasnim, a u radu dijela zaduženog za prikaz plana pokazala se potreba za mogućnošću promijene vizuelnih atributa prikaza tijekom rada programskog sustava. Tijekom testiranja zaključeno je da implementacija modela uveliko ovisi o programskoj platformi i upravitelju prozora, jer unatoč određenoj razini standardizacije dijelova i načina komuniciranja s grafičkim korisničkim sučeljima upravljačkih sustava, svaki sustav prozora posjeduje dodatne skupove biblioteka koji mogu olakšati i poboljšati izradu prototipa modela.

Na osnovu iskustava stečenih tijekom realizacije i testiranja prototipne verzije opisanog modela uočena je potreba za implementiranjem metoda umjetne inteligencije te razvojem jezika i pravila komuniciranja između dijelova sustava. U daljnim fazama razvoja modela korisničkog sučelja predviđena je i podrška mrežnom komuniciranju između dijelova programskog sustava.

LITERATURA

[1.] Finger S., Dixon J.R., A Review of Research in Mechanical Engineering Design. Part I: Descriptive, Prescriptive and Computer-Based Model of Design Process, Research in Engineering Design, Vol. 1, No. 1, 1989., 51-67

[2.] Finger S., Dixon J.R., A Review of Research in Mechanical Engineering Design. Part II: Representations, Analysis and Design for the Life Cycle, Research in Engineering Design, Vol. 1, No. 2, 1989., 121-137

[3.] Kostelić A., CAD kao podsustav CIM, Zbornik radova 4. skupa Konstruiranje 96, Tehnički fakultet Rijeka, 1996.

[4.] Marjanović D., Model prikaza konstrukcijskog procesa, Zbornik radova 4. skupa Konstruiranje 96, Tehnički fakultet Rijeka, 1996.

[5.] Ulbin M., Flašker J., Some Aspects of the Modern Design Techniques, 4. skup KONSTRUIRANJE 96 (Opatija), Publisher, Rijeka, 1996., 123-130

[6.] Marjanović D., Milošević V., Razvoj modela ICAE sustava, Zbornik radova FSB XVI (1992), Zagreb,1992., 108-115

[7.] Pavković N., Kreiranje baze scenarija ekpertnog sustava, Magisterij, FSB Zagreb, 1996.

[8.] Kostelić A., Marjanović D., General Design Theories - Pro Et Contra, International Conference on Engineering Design, ICED 95, WDK, Heurista, 1995.

[9.] Marjanović D., Implementacija ekspertnih alata u procesu konstruiranja, Disertacija, FSB Zagreb, 1995.

[10.] Deković D., Sustav za vođenje konstrukcijskog procesa, Diplomski rad, FSB Zagreb, 1985.

[11.] Salopek D., Sustav za objašnjavanje konstrukcijskog procesa, Diplomski rad, FSB Zagreb, 1996.

[12.] Marjanović D. Smojver B., Jović M., Bojčetić N., A Tool for Design of

Mechanical Power Transmision, The Europien CAD/CAM Conference, Münich, 1991.
[13.] Jović M., Sistem za automatsko modeliranje sklopova, Zbornik radova FSB XVI (1992), Zagreb, 1992., 157-165
[14.] Marjanović D., Pavković N., The Structure of an ICAE System, International Conference on Engineering Design, ICED 95, WDK, Heurista, 1995.
[15.] Bojčetić N., Jović M., The Model of User Interface in design Preocess, International Conference on Engineering Design, ICED 95, WDK, Heurista, 1995.
[16.] Marjanović D., Bojčetić N., Jović M., Pavković N., Smojver B., Struktura modela ICAE sustava, Zboranik radova FSB XVII (1993), Zagreb, 1993., 91-99
[17.] Jović M., Sustav za automatsko modeliranje sklopva, Rukopis magistarskog rada, FSB, Zagreb 1996.
[18.] Kostelić A., Znanost o konstruiranju, rukopis knjige, Zagreb, 1996.
[19.] Oberšmit E., Nauka o konstruiranju, metodičko konstruiranje i konstruiranje pomoću računala, SNL Liber, Zagreb, 1985.
[20.] Eliens A., Object-Oriented Software Development, Adison-Wesley, Cambridge, 1994.
[21.] Hubka V., Eder W.E., Engineering Design - General Procedural Model of Engineering Design. Heurista, Zürich, 1992.
[22.] Hubka V., Principles of Engineering Design, Heurista, Zürich, 1988.
[23.] Ullman G.D., The Mechanical Design Process, McGraw-Hill, New York, 1992.
[24.] Hubka V., Eder W.E., Theory of Tehnical Systems, Springer, Kingston, 1988.
[25.] Ullman G.D., Stauffer A.L., Fundamental Processes of Mechanical Designers Based on Empirical Data, Journal of Engineering Design, Vol 2. No. 2, 1991., 113.-125.
[26.] Bojčetić N., Model komuniciranja konstruktor računalo, Zboranik radova FSB XVI (1992), Zagreb, 1992., 149-157
[27.] Deković D., Marjanović D., Bojčetić N., Model sustava za vođenje konstrukcijskog procesa, Zbornik radova 4. skupa Konstruiranje 96, Tehnički fakultet Rijeka, 1996.
[28.] Milošević V., Konstrukcija vratila sa upotrebom ES, Ekspertni sistemi v cad procesu, seminar, Kralj, Ljubljana, 1990., 108.-117.
[29.] Kostelić A., Osnivanje ekspertnih sustava, Ekspertni sistemi v cad procesu, seminar, Kralj, Ljubljana, 1990., 26.-39.
[30.] Parsaye K., Chignell M., Expert Systems for Experts, Wily, New York, 1988.
[31.] Grimes D.J., Human Factors-Part 2, Computer Graphics and Applications, Vol. 4, No. 12, 1984.,10.-13.
[32.] Eberts E.R., User Interface Design, Prentice Hall, Englewood Cliffs, 1994.
[33.] Borufka H.G, Hannusa H., Weber H.R., Interactive Graphics Systems, Technische Hohschule Darmstadt, Darmstadt, 1983.
[34.] Peddie J., Graphical User Interfaces and Graphics Standards, McGraw-Hill,New York, 1992.
[35.] Nielsen J., Iterative User-Interface Design, Computer, Vol. 26, No. 11,1993.,

32-43.
[36.] Dam A., Marcus A., User-Interface Developments for Nineties, Computer Graphics and Applications, Vol. 24, No. 9, 1991.,49.-58.
[37.] Stroustrup B., The C++ Programming Language, Addison-Wesley Publishing Company, Reading, 1986.
[38.] Ege R., Object-Oriented Programming With C++, AP Professional, Cambridge, 1994.
[39.] Barr A., Feigenbaum A.E., The Handbook of Artificial Intelligence Volume II, HeurisTech Press, Stanford, 1982.
[40.] Dolphin C., Craufurd P., Bergan M., Alty L.J., Experiments using multimedia interfaces in process control: Some initial results., Computer&Graphics, Vol. 17, No. 3, 1993., 205.-219.
[41.] Väänänen K., Interfaces to hypermedia:Communicating the structure and interaction possibilities to the users, Computer&Graphics, Vol. 17, No. 3, 1993., 219.-229.
[42.] Java Unleashed, Sams.net, Indianapolis, 1996., (grupa autora)
[43.] Lemay L.,Perkins C. L., Teach Yourself Java in 21 Days, Sams.net, Indianapolis,1996.
[44.] Barr A., Feigenbaum A.E., The Handbook of Artificial Intelligence Volume I, Addison-Wesley Publishing Company, Reading
[45.] Gaskins T., PHIGS Programming Manual,O'Reilly & Associates, Inc., Sebastopol, 1992.
[46.] Puk F.R., McConnell I.J., GKS-3D: A Three-Dimensional Extension to the Graphics Kernal System, IEEE Computer Graphics an Applications, Vol. 6, No. 8, 1986.,42.-49.
[47.] Brodile W.K., GKS-94: An Overview, IEEE Computer Graphics and Applications, November 1995., 64.-71.
[48.] Young J. M., Mastering Visual C++, Sybex, Alameda, 1996.
[49.] Dehlin J.P., Curland J.M., Object Programming with Visual Basic4, Microsoft Press, Redmond, 1996.
[50.] Jagannathan V., Dodhiawala R., Baum S.L., Blackboard Architectures and Applications, Academic press, London, 1989.
[51.] Craig D.L., The Cassandra Architecture, Ellis Horwood, New York, 1989.
[52.] Ho C-S., Keng C-H, Development of a generalized knowledge source strucure for blackboard system, Konwledge-Based Systems, Vol. 7 No. 1, 1994, 5.-16.
[53.] Sydenham P., Sunners M., Building an interactive blackboard framework, Konwledge-Based Systems, Vol. 7., No. 3., 1994., 199.-205.
[54.] Hutchinson S.E., Sawyer S.C., Computers and Information Systems, Irwin, Boston, 1994.
[55.] Lipschutz S., Data Structures, McGraw-Hill, 1986.
[56.] Shildt H., C The Complete Reference, McGraw-Hill, New York, 1990.
[57.] Vally J., UNIX Programer's Reference, QUE, Carmel, 1991.

ŽIVOTOPIS

Rođen sam 26. 10. 1965. godine u Sisku. Osnovnu školu sam završio u Petrinji, a srednju (Centar za usmjereno obrazovanje Norebert Veber) u Sisku. Fakultet strojarstva i brodogradnje upisao sam 1985. godine. Diplomirao sam 1991. godine, na usmjerenju "Strojarske konstrukcije" kod prof. dr. Aurela Kostelića.

U lipnju 1991. godine primljen sam na Fakultet strojarstva i brodogradnje kao mladi istraživač na projektu broj 2-08-173 "Model inteligentnog CAE sustava", 1993. godine promijenjen mi je status u mađeg asistenta za znansveno područje strojarstvo u Zavodu za mehaničke konstrukcije, katedra za elemente i konstrukcije. Aktivno sudjelujem u nastavi na kolegijima Uvod u računala, Odabrane metode konstruiranja, Osnove konstruiranja pomoću računala i Informatika. Koristim se engleskim jezikom. Ukupno sam do sada objavio kao autor ili koautor 5 znanstvenih radova.