

1. Od računala do objektnog programiranja



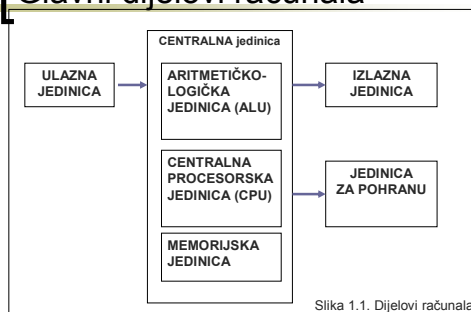
Razvoj poslovnih aplikacija, M.Zekić-Sušac

Povijesni razvoj programiranja

- Računalo - uređaj koji može izvoditi izračunavanja i donositi logičke odluke brzinom milijunima i milijardama većom nego što to mogu ljudi
- Računala procesiraju podatke na način da obradom upravljaju skupovi instrukcija nazvani RAČUNALNI PROGRAMI.
- Trend: troškovi hardvera stalno u opadanju, dok troškovi razvoja softvera i dalje rastu.
 - Metode razvoja programskih aplikacija omogućuju smanjenje tih troškova.

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Glavni dijelovi računala



Slika 1.1. Dijelovi računala

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Evolucija računalstva

- rana računala mogla su obraditi samo jedan zadatak u isto vrijeme – tzv. *Batch processing* (računalo obrađuje samo jedan program istovremeno, a podatke obrađuje u skupinama – batches)
- s razvojem računala i povećanjem njihovih mogućnosti, razvila se potreba da se više zadataka i poslova obavi u isto vrijeme dijeljenjem resursa računala – tzv. *Multiprogramming ili multitasking* (računalo dijeli svoje resurse na više zadataka, koji se međusobno „natječu“ čiji će se dio prvi obraditi, ali je ta obrada bila spora)
- 1960-tih – pojavljuju se operacijski sustavi koji su imali svojstvo *timesharing-a*.
 - *Timesharing* je vrsta multiprogramming-a kod koje korisnici pristupaju računalu putem terminala, svima se programi obrađuju na tom centralnom računalu na način da računalo pokreće mali dio korisnikovog zadatka, zatim se seli na dio drugog zadatka itd. To se obavlja velikom brzinom (dio svakog zadatka se obavi više puta u sekundi) i čini se gotovo simultano.

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Evolucija računalstva - nastavak

- 1970-tih (točnije 1977.) Apple popularizira pojam *osobnog računala*
- 1981. – IBM predstavlja IBM PC – preko noći računala postaju široko upotrebljavana u poslovanju, industriji, te javnoj upravi (PC računala su još uvijek „standalone“, pa nisu imala veliku moć obrade)
- PC računala se spajaju u lokalne mreže (LAN) ili putem telefonske mreže – pojavljuje se fenomen distribuiranog računalstva (*distributed computing*).
 - Distribuirano računalstvo je takvo kod kojeg se obrada podataka ne vrši na jednom centralnom računalu, nego na računalima korisnika u mreži (nastaje centralni odjel za obradu podataka u organizacijama)
- Javlja se koncept klijent – server (korisnik – poslužitelj) računalstva, pri čemu neka računala (poslužitelji) svoje resurse dijele drugim računalima u mreži (korisnicima)

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Evolucija programskih jezika

Programi su skupovi instrukcija pisani u različitim programskim jezicima, od kojih neke računalo direktno razumije, dok su za druge potrebni programi prevoditelji.

Danas se koristi na stotine programskih jezika, a mogu se podijeliti u tri osnovne skupine:

- strojni jezici
- asembleri
- jezici više razine

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Strojni jezik

Strojni jezik – „prirodni“ jezik računala, definiran dizajnom hardvera nekog računala.

Sastoji se od niza brojeva (koji se mogu reducirati na 0 i 1) koji instruiraju računalo da obavlja svoje osnovne elementarne zadatke. Strojni jezici su ovisni o računalu, tj. jedan jezik se može koristiti samo na tom tipu računala.

Primjer programa u strojnom jeziku koji zbraja prekovremenu plaću i osnovnu plaću, te rezultat memorira u ukupnu plaću (Deitel, Deitel, 1994):

```
+1300042774
+1400593419
+1200274027
```

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Asemblerski jezici

Umjesto korištenja isključivo brojki, ovi jezici nastoje ubrzati programiranje korištenjem skraćenice engleskog jezika koje predstavljaju osnovne računalne operacije. Za prijevod ovih skraćenica na strojni jezik koriste se prevoditelji, koji se za ove jezike zovu asembleri.

Primjer programa u assembleru koji također zbraja prekovremenu plaću i osnovnu plaću, te rezultat memorira u ukupnu plaću (Deitel, Deitel, 1994):

```
LOAD BASEPAY
ADD OVERPAY
STORE GROSSPAY
```

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Viši programski jezici

Viši programski jezici – ubrzavaju proces programiranja na način da se jednom instrukcijom mogu izvršiti neki značajni zadaci. Programi prevoditelji koji prevode više programske jezike u strojni zovu se kompajleri (compilers). Ovi jezici osim eng. riječi u naredbama koriste i matematičke i logičke operatore.

Primjer programa u višem programskom jeziku (Basic-u) koji također zbraja prekovremenu plaću i osnovnu plaću, te rezultat memorira u ukupnu plaću (Deitel, Deitel, 1994):

```
Grosspay = basePay + overTimePay
```

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Interpreteri vs. compiler-i

- Kako je proces prevodenja (kompajliranja) cijelog programa s višeg programskog jezika u strojni dugotrajan, uvedeni su i **interpreteri** – prevoditelji koji mogu direktno izvršiti programske jezike više razine bez potrebe za prevodenjem u strojni jezik.
- Razlika u funkcionalnosti:
 - kompajlirani programi se brže izvršavaju od interpretiranih programa,
 - interpreteri se koriste u razvojnim okolinama gdje se često dodaju nove mogućnosti programu, pa je potrebno često prevodenje i ispravljanje grešaka. Kada se program (aplikacija) jednom razvije, kreira se kompajlirana inačica koja će se brže izvršavati.

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Podjela viših programskih jezika

Viši programski jezici mogu se podijeliti u dvije skupine:

- PROCEDURALNI (strukturni)
- OBJEKTNI

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Strukturni programski jezici

Strukturni programski jezici su takvi koji se program izvodi slijedeći instrukcije jednu za drugom.

U prošlosti su među najraširenijim jezicima bili:

- Fortran (FORMula TRANslator) – razvijen 1954. od strane IBM-a za upotrebu u znanstvenim i inženjerskim aplikacijama koje zahtijevaju složene matematičke proračune. Još uvijek se koristi za inženjerske aplikacije.
- Cobol – (Common Business Oriented Language) – razvijen 1959 od strane grupe proizvođača računala, te korisnika u javnom sektoru i industriji. Koristilo se primarno za komercijalne aplikacije koje su zahtijevale precizne i učinkovite obrade velikih količina podataka.
- C – razvijen 1967 (autor: Martin Richards) na temeljima BCPL i B jezika s namjenom pisanja operacijskih sustava i prevoditelja. To je bio jezik koji je dodao tipove podataka i druge mogućnosti. Postao je poznat kao jezik u kojem je pisan UNIX operacijski sustav. U kasnim 1970-tim godinama razvija se „klasični C“ jezik (autori: Kernighan i Ritchie). Široka upotreba ovog jezika dovela je do razvoja više inačica za različite tipove računala, pa je bilo potrebno uvesti standardizaciju s ciljem da se C može koristiti neovisno o tipu računala i platformi. Rezultat tih udruženih napora od strane American National Standards Committee i ISO je ANSI C, koji je bio široko upotrebljavan proceduralni jezik.
- Pascal – Dizajniran u približno isto vrijeme kada i C (autor: Niklaus Wirth), bio je namijenjen za akademsku upotrebu

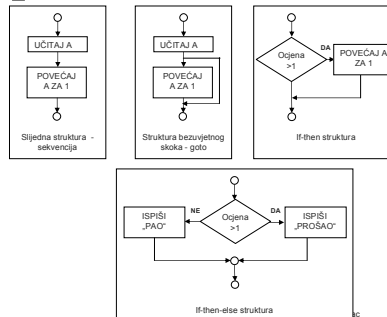
Razvoj poslovnih aplikacija, M.Zekić-Sušac

Principi strukturnog programiranja

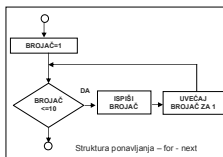
- Karakteristike: jasniji i pregledniji od nestrukturiranog, lakši za testiranje i otklanjanje grešaka (debug), te za modificiranje.
- Kod takvog se programiranja koristi dogovoreni skup struktura:
 - **Sekvencija** (ili slijedna struktura) pri čemu se naredbe izvršavaju jedna za drugom
 - **Bezuvjetni skok (goto)** – slanje programa na određenu liniju koda
 - **Selekcija** – ispitivanje uvjeta (if – then – else) – izbor između dvije različite akcije (ili između izvršavanja neke akcije i ignoriranja) koji je određen time da li je uvjet zadovoljen ili ne
 - **Ponavljanje (repeticija)** – strukture do-while i for (ponavljanje nekih naredbi sve dok je uvjet zadovoljen ili sve dok neki brojač ne dostigne određenu vrijednost)

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Strukture



Strukture - nastavak



Struktura ponavljanja – for - next

Slika 1.2. Strukture u struktornom programiranju

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Zašto objektno programiranje?

- S vremenom je bilo potrebno kreirati softver koji će se izgraditi što brže, točnije i ekonomičnije. Kao revolucija u programiranju javlja se uvođenje OBJEKATA kao softverskih komponenti koje se mogu više puta koristiti (reusable), a koje modeliraju elemente iz stvarnog svijeta. Kreatori softvera (developeri) su otkrili da se korištenjem modularnog, objektno-orijentiranog (OO) dizajna i pristupa implementaciji postiže veća produktivnost s pomoću strukturiranog programiranja.

Prednosti objektnih jezika:

- lakši za razumijevanje jer su bliži stvarnom svijetu
- lakši za ispravljanje i modificiranje

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Prednosti OO programiranja

Objektno orijentirano programiranje je jedan od najzastupljenijih smjerova u programiranju danas.

Prednosti:

- OO programiranje pojednostavljuje razvoj i testiranje iznimno velikih i kompliciranih sustava.
 - Sadrži mogućnosti apstrakcije podataka, enkapsulacije, polimorfizma i nasljeđivanja.
- .NET platforma je u potpunosti dizajnirana za razvoj objektno orijentiranih aplikacija i kao takva je odlična podloga za učenje i prihvaćanje OO strategije u razvoju modernih aplikacija.

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Primjeri OO programskih jezika

Među prvim objektnim jezicima bili su:

- SmallTalk (razvijen u Xeroxu),
- C++ (koji je zapravo hibridni jezik, jer je u njemu moguće koristiti i sve naredbe C jezika, ali i dodatne objektno-orijentirane). C++ je postao dominantnim programskim jezikom 1990-tih godina.

Danas su često korišteni objektni jezici: Java, C#, C++, Visual Basic, Python, i dr.

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Windows programiranje

Što je Windows?

- „Grafički operativni sustav koji za prikaz svojih informacija koristi grafičko korisničko sučelje... Temelji se na prozorima (windows), događajima, te ugrađenim objektima poput prozora za otvaranje datoteka.“ (Hladni, 2003)

Povijesni pregled:

- 1985. – prvi javno objavljen Windows operativni sustav v.1.0
- 1987. – razvijena verzija v.2.0
- 1990. – razvijena verzija v.3.0 s kojom Windows postaje najpopularniji operativni sustav

Razvoj poslovnih aplikacija, M.Zekić-Sušac

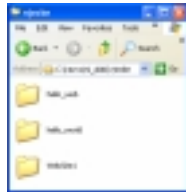
Odluke Windows op. sustava

- U operativnim sustavima koji koriste grafičko korisničko sučelje jednostavan je rad sustavom zbog mogućnosti korištenja slikovnih izbornika čije se naredbe i opcije mogu odabirati i aktivirati mišem. Sve informacije na zaslonu se tretiraju kao slika te se kopiranjem mogu prebaciti u drugi program, a također je na zaslonu omogućen WYSIWYG prikaz (eng. what you see is what you get) identičan onome koji će se vidjeti ako se slika ili dokument ispiše na papiru.
- Važna odlika operativnog sustava Windows je da **objekte ugrađene u operativni sustav može koristiti svaka aplikacija** koja radi na tom operativnom sustavu. Ta karakteristika dovodi do dvojakе prednosti:
 - za korisnike - olakšano učenje upotrebe novih aplikacija (jer je korisničko sučelje različitih aplikacija zapravo vrlo slično, opcije se koriste na sličan ili isti način)
 - za programere – upotrebom gotovih objekata ugrađenih u operativni sustav vrijeme izrade aplikacija se drastično smanjuje

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Prozor u Windows op. sustavu

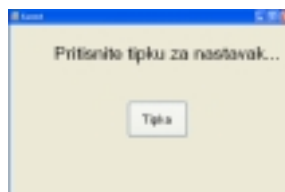
- Prozori su pravokutni dijelovi zaslona koji imaju svoju veličinu, mogu se otvoriti ili zatvoriti, te imaju svoj identifikacijski broj unutar operativnog sustava – window handle.
- Primjer nekih prozora: svi gumbi (uključivši i gumb Start), ikone na traci sa zadacima (Task Bar-u), okviri za unos teksta, okviri za rad s aplikacijom i dr.



Slika 1.3. Primjer prozora u Windows operativnom sustavu

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Događaji i poruke



Windows aplikacija većinu svog vremena mirno čeka da Vi kao korisnik pokrenete neki događaj (npr. da aktivirate neku naredbu). Događaj može nastati klikom miša, pritiskom na tipku, pomicanjem prozora na zaslonu, ili dr.

Pokrenuti događaj generira poruku koja dolazi do aplikacije, i ona odgovara na tu poruku. Ta se poruka proslijeđuje drugim prozorima. Poruka je, dakle, sredstvo za proslijeđivanje informacija prozorima o događaju koji se zbilo.

Ako želimo da se nakon slanja poruke nešto dogodi, potrebno je u programu napisati **odgovor na događaj** (event procedure).

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Programiranje odgovora na događaj

Programiranje Windows aplikacija može pojednostavljeno opisati kao **odgovaranje na događaje**.

- Da li se u nekoj Windows aplikaciji mora pisati odgovor na svaki događaj? Ne, jer na neke događaje odgovara sam operativni sustav, npr. ako korisnik aplikacije aktivira kombinaciju tipaka Alt+F4 operativni sustav će zatvoriti prozor aplikacije bez potrebe da programer piše programski kod za taj događaj.
- U slučaju kada operativni sustav nema odgovor na neki događaj, tada se poruka o događaju proslijeđuje otvorenoj aplikaciji. Ukoliko je programer aplikacije isprogramirao odgovor na taj događaj, isti će se izvršiti, a ukoliko nije, neće se dogoditi ništa.

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Objekti u OO programiranju

- Osim što neka Windows aplikacija odgovara na događaje, ona i rukuje određenim **objektima**, npr. prozorima, gumbima, okvirima za tekst, oznakama, slikama, i dr.

Što su objekti kod OO programiranja?

- Objekti u programskim jezicima predstavljaju objekte iz stvarnog svijeta. Objektima se pridružuju:
 - svojstva (eng. properties),
 - postupci i
 - odgovori na događaje.

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Pridruživanje svojstava objektima i pokretanje postupka

Pridruživanje svojstava objektima vrši se na način da se navede ime objekta, zatim točka, ime svojstva, te na kraju pridružena vrijednost svojstva,

- npr. ako želimo reći da je neki student visok 170 cm, tada bismo to u objektnom programu napisali ovako:
`Student.Visina=170`
- ako želi tom studentu reći da **pokrene postupak** (ili **metodu**) čitanja, tada bismo to napisali ovako:
`Student.Citaj`

← svojstvo (property)

← Postupak (metoda)

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Kontrole u OO programiranju

Kao objekti u objektnom programiranju često se koriste **kontrole** za rukovanje aplikacijom, npr. kontrola Label, Textbox, Listbox, Button i dr.

Kontrole možemo po potrebi umetati u svoju aplikaciju, a sve one imaju neka zajednička svojstva, kao npr. Name (naziv kontrole), Enabled (da li će ta kontrola biti omogućena za korištenja ili ne), Visible (da li će kontrola biti vidljiva ili ne), Text (koji će se tekst upisati u toj kontroli) i dr.

- Npr. ukoliko želimo u prozoru aplikacije ispisati poruku „Zdravo, svijete!“, to možemo napisati ovako:

```
label1.Text="Zdravo, svijete!"
```

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Programiranje odgovora na događaj

Ukoliko želimo isprogramirati odgovor na neki događaj, potrebno je u aplikaciju umetnuti kontrolu koja će generirati događaj (npr. gumb na koji će korisnik kliknuti mišem – Button kontrolu).

Ako prilikom dizajniranja aplikacije dvaput klinkete mišem na taj gumb, definirali ste „klik“ događaj (koji je najčešći događaj u objektnim aplikacijama), a u C# jeziku će se automatski generirati sljedeći kod:

```
private void button1_Click(object sender, EventArgs e)
{
    label1.Text = "Zdravo, svijete!";
}
```

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Button1_Click procedura

Naredba private void **button1_Click** označava proceduru (potprogram) koja će biti pozvana i izvršena svaki put kada korisnik klikne na taj gumb, odnosno kada aktivira događaj **button1_Click**.

Iza naziva procedure u okrugloj zagradi se navode argumenti, a naredbe koje se izvode u sklopu procedure navedene su u vitičastim zagradama.

U gornjem primjeru u proceduri se izvršava samo jedna naredba koja će u prozoru aplikacije ispisati poruku „Zdravo, svijete!“.

Sve naredbe jedne procedure obično se pišu uvučeno u odnosu na naziv procedure, radi preglednosti programskog koda.

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Glavne aktivnosti za programiranje Windows aplikacije

Iz navedenog se može zaključiti da su glavne aktivnosti za pisanje Windows aplikacije:

- definiranje glavnih klasa i objekata, te dodavanje kontrola
- mijenjanje svojstava tih kontrola
- pisanje odgovora na događaje

Razvoj poslovnih aplikacija, M.Zekić-Sušac

Razine programiranja

- Programiranje aplikacija radne površine (“desktop”) aplikacija – za poslovne aplikacije, video igre, aplikacije uredskog poslovanja i dr.
- Sistemske programiranje – izrada programa koje op. sustav koristi u pozadini za zadatke na sistemskoj razini
- Razvoj web aplikacija – izrada aplikacija kreiranih za izvršavanje u preglednicima (browserima), moguće osim prog. jezika koristiti i html i skriptne jezike (php, asp i dr.)

Razvoj poslovnih aplikacija, M.Zekić-Sušac

[Literatura]

- Deitel, H.M., Deitel, P.J. C++ How to Program, Prentice Hall, Englewood Cliffs, New Jersey, 1994.
- Hladni, I., Visual Basic .NET ab ovo, Miš, Zagreb, 2003.
- J. Liberty, Programming C#, Fourth edition, O'Reilly and Associates, 2005.
 - Prijevod: C# programiranje, A. Dragosavljević (ur.), Dobar plan, 2005.
- S. Barker, Visual C# 2005 Express, Mihailo J. Šolajić (ur.), Kompjuter biblioteka, 2007.
- Predavanja i zadaci na web stranici kolegija
- Web izvori



Razvoj poslovnih aplikacija, M.Žekić-Sušac