Softver

Programi – softver:

 nematerijalna komponenta računala

 niz instrukcija unaprijed pripremljenih i pohranjenih u radnoj memoriji računala tako da ih procesor može izvoditi

 Postupak rješavanja određenog problema ugrađuje se u računalo u obliku računalnog programa

 Programe piše čovjek koristeći se umjetno stvorenim jezicima za komunikaciju čovjeka i računala

Kategorije softvera:

 sistemski (sustavski)

 pokreće računalo

 upravlja računalom

 aplikativni

 rješava određen zadatak odnosno problem korisnika

Sistemski softver:

 aplikativni softver ne može raditi bez sistemskog softvera

 sistemski softver:

 pokreće računalo (“booting”)

 interpretira instrukcije i podatke

 upravlja računalom (hardverom)

 operacijski sustavi

 pomoćni programi

 programi prevoditelji

Operacijski sustav (OS):

 upravlja svim strojnim komponentama računala

 omogućuje izvršavanje korisničkih aplikacija

 interne/eksterne upravljačke instrukcije

 upravljački program je stalno u memoriji i upravlja svim ostalim programima

Programski jezici:

Programi prevoditelji:

 računalo izvodi strojne instrukcije u strojnom jeziku, koje su čovjeku nerazumljive

 čovjek programira u simboličkim ili programskim jezicima

Strojni jezik – 1. gen.:

 instrukcije i podaci pišu se binarno binarni niz 10001011 - naredba za zbrajanje

 ovisan o računalu - procesoru

 strojne instrukcije su jednostavne

 obavljaju osnovne aritmetičke i logičke operacije

 složene operacije moraju se obaviti kombiniranjem osnovnih operacija (programiranjem !)

prvi kratki korisnički program pohranjen u računalo

“the Baby” - University of Manchester, 1948.

Asembler – 2. gen.:

 jedna naredba asemblera odgovara jednoj naredbi strojnog jezika

 mnemonički jezik kojim se binarna naredba strojnog jezika zamjenjuju ključnom riječi naredba za zbrajanje:

10001011 (strojni jezik) = ADD (asembler)

Asembler

 prevođenje na strojni jezik obavljaju programi prevoditelji (sklapati, engl. to assemble)

Viši programski jezici – 3. gen.:

 programiranje sliči pisanju formula ili rečenica

 jedna naredba prevodi se u više naredbi (5-10) strojnog jezika

 prevođenje obavljaju programi prevoditelji:

 kompilatori (compilers)

 interpreteri

 jezici su općeniti, neki su i standardizirani pa su programi neovisni o računalu

 proceduralni jezici

 postupak rješenja problema opisuje se nizom programskih naredbi - procedurom (KAKO riješiti problem)

 primjeri: Basic, Fortran, COBOL, Pascal, C

Viši programski jezici – 4. gen.:

 znatno efikasniji od jezika 3. gen.

 jedna naredba pretvara se u 10-100 naredbi strojnog jezika

 programeri su znatno produktivniji

 većina jezika je neproceduralna

 ŠTO treba riješiti

 upotreba

 upitni jezici

 izvještajni jezici

 generatori aplikacija viši programski jezici – 4. Gen.

 primjeri:

 QBE – Query By Example,

 SQL – Structured Query Language

Aplikacijski softver:

 svi korisnički programi kojima se rješavaju specifični problemi korisnika

 Paketi aplikacijskog softvera

 aplikacije opće namjene (MS Office)

 aplikacije specifične namjene (glazba, sport, matematika..)

 opće poslovne aplikacije (upravljanje zalihama, računovodstveni paketi, financije, kadrovska evidencija..)

 specifične poslovne aplikacije (građevinarstvo, arhitektura, medicina, turizam, ugostiteljstvo..)

 aplikacije na području obrazovanja (CAL- Computer Assisted Learning, CAI-Computer Aided Instruction, MS Encarta..)

 kućne aplikacije - igre, zabava, kućne financije

Kakve sve vrste aplikacija postoje i kako ih organizacije dobavljaju?

 Aplikacijski softver

 Obavlja uslugu ili funkciju

 Neki su općeniti, kao što su Microsoft Excel ili Word

 Drugi pružaju specifične funkcije, kao što je knjigovodstvo kroz softver Adacco - domaći poslovni

i knjigovodstveni softver.

Kakve kategorije aplikacijskog softvera postoje?

 Horizontalne aplikacije

 Mogu ga koristiti organizacije različitih djelatnosti. Obrada teksta, grafički programi, tablični kalkulatori, prezentacijski programi su primjeri horizontalnog aplikacijskog softvera.

 Primjeri:

 Microsoft Word, Excel, PowerPoint; Adobe Acrobat, Photoshop, PageMaker…

 Koriste ih različiti poslovni subjekti iz različitih industrija

 Kupuje se u gotovoj verziji (off-the-shelf) i prilagodbe su moguće u vrlo maloj mjeri (ako uopće)

 Vertikalne aplikacije

 Koristi se u određenoj industriji / djelatnosti

 Primjeri:

 Programi za zubarske ordinacije u kojima se mogu bilježiti sastanci sa pacijentima; automehaničarske radionice za vođenje evidencije dijelova na zalihama i programi za vođenje zaliha u maloprodaji.

 Obično se mogu prilagoditi pojedinom korisniku. Tipično će dobavljač softvera ponuditi uslugu kvalificiranog konzultanta koji će prilagoditi softver.

 Specifične aplikacije

 Razvijaju se za specifične, jedinstvene potrebe

 Primjer – ISVU

Firmware:

 Računalni softver koji je već instaliran u opremu kao što su printeri, print serveri i različita mrežna oprema.

 Programira se kao i ostali softver, ali se instalira na specijalnu, read-only memoriju na printer ili drugu opremu, tako da softver postaje dio memorije uređaja.

 Korisnici ne trebaju posebno brinuti o ovom softveru.

 Može se nadograditi ili zamijeniti, ali to je obično zadatak informatičkog stručnjaka. Zadatak je jednostavan, ali zahtijeva specijalistička znanja koja obično nisu u fokusu interesa ekonomista.

Softverska ograničenja

Važna softverska ograničenja:

1. Određena verzija operacijskog sustava napisana je za posebnu vrstu hardvera.

 Na primjer, Microsoft Windows radi samo na Intel procesorima ili na procesorima koji rade sukladno Intel instrukcijskom setu (naredbama koje upravljaju procesorom).

 32-bitna verzija Windowsa radi samo na Intel 32-bitnim procesorima (ili kompatibilnim), a 64-bitna verzija Windowsa radi samo na Intel 64-bitnim procesorima (ili kompatibilnim).

 Linux koristi drugačije instrukcije i može se pokretati i na 32- bitnim i na 64-bitnim računalima.

2. Aplikacijski programi napisani su za određene operacijske sustave.

 Microsoft Access može raditi samo na Windows operacijskom sustavu.

 Neke aplikacije mogu imati više verzija.

 Windows i Macintosh verzija Microsoft Worda.

 Ako nemate drugačije informacije, pretpostavite da aplikacija radi samo na jednom operacijskom sustavu.

Softver otvorenog koda:

Open Source pokret i proces

 GNU (Gnu Not Unix) – jedan od prvih softvera otvorenog koda

 GNU General Public License (GPL) Agreement

 Jedan od standardnih licencnih ugovora za softver otvorenog koda

 Uspješni projekti otvorenog koda

 OpenOffice (sličan Microsoft Office-u)

 FireFox (preglednik)

 MySQL (DBMS)

 Apache (Web server)

 Ubuntu (operacijski sustav kao Windows)

 Android (softver za mobitele)Open Source pokret i proces komercijalni

Zašto programeri volontiraju u radu sa softverom otvorenog koda?

 Kroz takve aktivnosti mogu naučiti mnogo, a i zabavno je…

 Imaju slobodu raditi na projektu na kojem god to žele

 Mogu iskazati svoju kreativnost dok rade na projektima koji su im zanimljivi i korisni

 Nauče dobro određene vještine, zbog osobnog ponosa, ali i kao prednost kod zapošljavanja

 Mogu osnovati ili raditi u poduzeću koje pruža usluge vezane uz softver otvorenog koda

Razvoj softvera:

Kako organizacije nabavljaju softver?

 Kupovina gotovog softvera

 Prilagodba gotovog softvera

 Softver po posebnoj narudžbi

 Organizacije razvijaju aplikacijske softvere za sebe ili angažiraju za to specijalizirane tvrtke.

 Razvoj softvera po narudžbi je težak i rizičan.

 Zapošljavanje i upravljanje softverskih timova nije jednostavno.

 Upravljanje projektom za razvoj softvera može biti zahtjevno. Često takvi projekti traju dulje od očekivanog.

 Prebačaj troška od 200 do 300% nije rijedak.

 Ovu soluciju treba koristiti samo kada nema drugih opcija.

Proces razvoja softvera:

 Vodopadni model

 Slijedni pristup

 Slijedeća faza započinje po završetku prethodne faze

 Rezultat (softverska aplikacija) dostupna na kraju projekta

 Evolucijski model

 Kontinuirana interakcija korisnika i informatičara

 Na temelju početnih zahtjeva razvija se aplikacija (prototip)

 Aplikacija se razvija do konačne verzije koja zadovoljava potrebe korisnika

Razvoj softvera kao projekt:

 Izrada softvera → projekt

 Planski pristup, po određenoj metodi

 Definiranje projektnog zadatka

 Formiranje projektnog tima …

 Upravljanje softverskim projektom

 Praćenje (vođenje, nadzirenje) procesa razvoja softvera kroz sve faze njegovog razvoja

 Praćenje rokova kroz terminske karte (MS Project), definiranje kontrolnih točaka (miljokazi) projekta …

Razlozi neuspjeha u razvoju softvera:

 Improvizacija

 Korisnik nije aktivno uključen u projekt

 Dijelovi IS-a nisu usklađeni

 Raspoloživa oprema je neadekvatna

 Softver ne zadovoljava potrebe

 Nedovoljna educiranost članova tima

 Prekoračenje rokova ….

Faze razvoja vodopadnog pristupa:

Faze razvoja softverske aplikacije

putem vodopadnog pristupa

 Definiranje i analiza zahtjeva

Modeliranje aplikacije

Razvoj aplikacije

Testiranje

Verifikacija i validacija

Implementacija

 Održavanje

Definiranje i analiza zahtjeva:

 Definiranje zahtjeva

 Zahtjevi dolaze od:

 Korisnika

 Informatičara koji razvijaju sustav

 Okoline u kojoj će se softver primjenjivati

 Analiza zahtjeva

 Zahtjevi prihvatljivi → definiranje rokova, raspoloživih sredstava, potrebnih stručnjaka …

Modeliranje aplikacije:

 Vizualizacija problema nekom od tehnika modeliranja

 Klasične tehnike modeliranja

 Dijagram s blokovima

 Tablica odlučivanja

 Dijagram toka podataka

 Danas: UML (Unified Modelling Language)

 Akcijski dijagrami

Razvoj aplikacije:

 Aplikacija se može razvijati na način da se:

 u cijelosti programira u nekom programskom jeziku (program = niz instrukcija; instrukcija = operacija)

 u cijelosti razvija primjenom softverskog alata

 kombinira programiranje i korištenje alata

 koriste ranije razvijene i implementirane programske komponente

 Posebno važno – razvijati aplikaciju u skladu s definiranim zahtjevima

Testiranje:

 Kontinuirano, tijekom stvaranja aplikacije

 Testiranje:

 Pojedinih komponenata

 Manjih cjelina (modula)

 Integracijsko testiranje

 Testirati testnim podacima:

 Obuhvatiti sve putove i dijelove programa

 Posebno:

 Dijelove koje je bilo teško programirati

 Najjednostavnije dijelove programa verifikacija i validacija

 Verifikacija: Da li se dobro gradi softverski proizvod?

 U skladu s pravilima struke, sa specifikacijom poslova …

 Validacija: Da li se gradi dobar proizvod?

 Funkcionalno zadovoljavanje zahtjevima korisnika (statička i dinamička kontrola), testiranje uporabljivosti, pouzdanosti softvera.

Implementacija:

 Pripremne aktivnosti:

 Instalacije opreme

 Punjenje baze podataka

 Edukacija zaposlenika …

 Implementacija softvera:

 Paralelni način rada (stara i nova aplikacija)

 Prijelaz na novu aplikaciju

 Dokumentacija

 Tehnička dokumentacija

 Korisnička dokumentacija

Održavanje:

 Zahtjev za održavanjem dolazi od:

 Korisnika

 Informatičara

 Okoline

 Zahtjev mora biti:

 Dobro dokumentiran

 Podliježe analizi (opravdanost, tehnička izvedljivost)

 Svaka promjena se odobrava i dokumentira

 Održavanje softvera do granice isplativosti

Dobava softvera:

 Kupnja gotovog softvera

 MS Office

 Korisnički razvoj - korisničko računarstvo

 Aplikacije u Excelu, Accessu

 Vlastiti razvoj u informatičkoj službi

 informacijski sustavi

 Korištenje vanjske usluge (outsourcing)

Mala i srednja poduzeća:

 Obično standardiziraju klijentska računala, jer je održavanje više različitih klijentskih računala znatno kompliciranije i skuplje.

 Najčešće odabiru Microsoft Windows. Ako se bave umjetnošću ili dizajnom odabiru Macintosh, a ako se bave znanošću ili inženjerskim poslom odabiru Unix.

 Organizacije koje imaju ograničen budžet trebaju razmisliti o odabiru softvera otvorenog koda kao što je Linux, Ubuntu ili Open Office, ali to je ipak rjeđe u praksi.