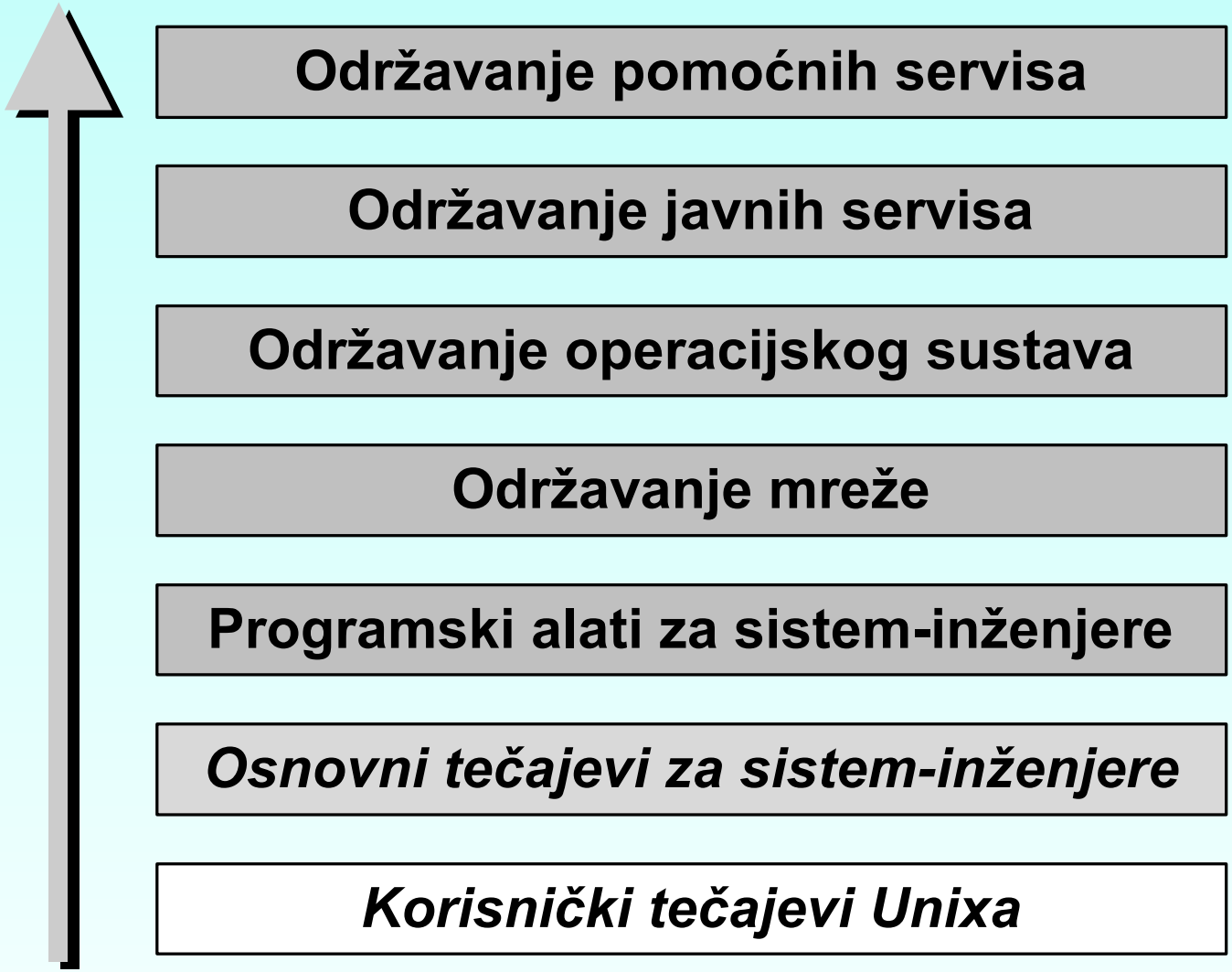
Programski alati

autor: Branko Radojević (@vdu.hr)

mentor: Dobriša Dobrenić (@srce.hr)
recenzent: Zdenko Škiljan (@srce.hr)


Održavanje pomoćnih servisa
Održavanje javnih servisa
Održavanje operacijskog sustava
Održavanje mreže
Programski alati za sistem-inženjere
Osnovni tečajevi za sistem-inženjere
Korisnički tečajevi Unixa

Ciljevi tečaja

- Prezentirati tipove programskih alata
- Upoznati se s različitim inačicama sistemskih ljuski (*shell*)
- Upoznati se s programskim jezikom *C*
- Dati osnove *Perla*
- Upoznati funkcioniranje *WWW/CGI* skripti
- Upoznati se s *PHP*-om

Potrebno predznanje

- Znanja sa Osnovnih tečajeva za sistem-inženjere
- Osnovno poznavanje *UNIX*-a
- Osnovno poznavanje shella
- Poznavanje instalacije i rada web poslužitelja (*Apache*)

Sadržaj

Interpreteri i prevodioci

Interpreteri

- Interpreter prima izvorni kôd skripte ili programa i prevodi ga liniju po liniju kôda u trenutku samog izvršavanja programa
- Kôd se obično distribuira u izvornom obliku koji se lako može izmijeniti i prilagoditi našim potrebama (prednost i mana)
- Kôd se može izvoditi na više računalnih platformi
- Kôd se sporije izvodi od prevedenog kôda

Interpreteri i prevodioci

Prevodioci

- Prevodilac zahtijeva prethodno prevođenje izvornog kôda u strojni kôd
- Kôd se može distribuirati u izvornom obliku (source code) koji tek treba prevesti na određenoj platformi, ili se može distribuirati u binarnom obliku za određenu računalnu platformu
- Kôd se brže izvodi od interpretiranog kôda

Interpreteri i prevodioci

Ljuska (shell)

- Osnovni alat svih *UNIX* sistem-inženjera
- Postoji više inačica ljuski na *UNIX*-u
- Osnovna *UNIX* ljuska je *Sh*
- Postoje još i *Bash*, *Csh*, *Tcsh*, *Ksh* ...
- Skripte pisane u ljusci se distribuiraju u izvornom kôdu i ljuska interpretira skripte liniju po liniju kôda
- Korisne su za manje zahvate i automatizaciju redovnih poslova održavanja poslužitelja

Interpreteri i prevodioci
Nawk

- Jezik za pretraživanje i obradu uzoraka (pattern scanning and processing language)
- Novija verzija *Awk*-a, na novijim inačicama operacijskih sustava zamjenjuje *Awk*, dopunjen brojnim novim mogućnostima
- Koristi se za automatiziranje složenijih poslova, najčešće kod zahvata na tekstualnim datotekama (npr. */etc/passwd*) i obrađivanja njihovog sadržaja
- Besplatan

Interpreteri i prevodioci
Perl

- Practical Extracting & Reporting Language
- Razvio ga Larry Wall, besplatan
- Zaokružen jezik koji je spoj najboljih osobina *C*-a, *Pascala*, *Awk*-a i *Basica*
- Spada u interpretere, iako prilikom prvog pokretanja prevodi izvorni kôd u svoj izvršni meta kôd
- Lako (i automatizirano) se nadopunjuje modulima

Interpreteri i prevodioci
PHP

- *PHP* je skriptni jezik koji se dade uključiti u poslužitelj web stranica (server-side, cross-platform, HTML embedded scripting language)
- Koristi se gotovo isključivo za automatizaciju i dinamičko generiranje web stranica
- Može se spajati gotovo na sve postojeće baze podataka (*PostgresSQL*,*Oracle*, *MySQL*, ...)
- Spada u interpretere
- Razvija ga *Apache Software Foundation*
- Besplatan

Interpreteri i prevodioci
Tcl/Tk

- *Tcl* (Tool Command Language) skriptni jezik s vrlo jednostavnom sintaksom
- *Tk* je nadogradnja na Tcl koja omogućava jednostavno i brzo programiranje grafičkog korisničkog okruženja (*GUI*)
- Može se upotrebljavati samostalno ili dinamičkim pozivanjem iz web stranica
- Spada u interpretere
- Besplatan

Interpreteri i prevodioci
Scotty

- Nadogradnja za *Tcl/Tk* koja je orijentirana prema upravljanju mrežnim servisima i poslužiteljima
- Proširuje mogućnosti *Tcl/Tk* skriptnog jezika i omogućuje uporabu *SNMP*, *ICMP*, *DNS*, *HTTP*, *SUN RCP*, *NTP* i *UDP* protokola
- Koriste ga proizvođaći mrežne opreme za automatizirano očitavanje statusa opreme putem *SNMP* protokola
- Besplatan

Interpreteri i prevodioci
Expect

- Alat za automatizaciju interaktivnih aplikacija kao što su *Telnet*, *Rlogin*, *Ftp* i sl.
- Može se koristiti za automatizaciju akcija koje treba povremeno raditi, npr. sistem-inženjer može napraviti skriptu koja se u određeno vrijeme može prijaviti na drugi stroj i napraviti Ftp-om arhiviranje (backup) tog stroja
- Uskladljiv s *Tcl/Tk*
- Spada u interpretere
- Besplatan

Interpreteri i prevodioci
Python

- Kombinira zavidnu snagu objektnog programskog jezika s jasnom sintaksom
- Smatra se interpreterom
- Može biti i interaktivan
- Često se koristiti i kao proširenje za dinamičko generiranje web stranica
- Postoji gotovo na svim platformama
- Besplatan

Interpreteri i prevodioci

Java

- Pokušaj stvaranja višeplatformnog okruženja za izvođenje programskog kôda
- Nastala u tvrtki *SUN*, u međuvremenu je prihvatile sve vodeće informatičke tvrtke u svijetu
- Microsoft posjeduje “svoju” verziju Jave koja nije 100% uskladiva sa Sunovom verzijom
- Brzinom izvođenja zaostaje za *C*-om, ali i nekim interpreterskim jezicima
- Dijelom se može smatrati interpreterom, a dijelom prevodiocem

Interpreteri i prevodioci

C

- Najkorišteniji programski jezik
- Postoji na svim računalnim platformama
- Većina operacijskih sustava je pisana upravo u *C*-u (*UNIX*)
- Pogodan za pisanje sistemskih programa
- Proširen podrškom za objektno programiranje: *C++*
- Spada u prevodioce
- *gcc* (*GNU C Compiler)* je besplatan

Sistemske skripte: Shell

Razlike i izbor

- *sh* - osnovna verzija - *Bourne shell* ($)
  postoji na svim inačicama *UNIX*-a
- *bash* - napredna verija sh - *Bourne again shell* ($)
- *csh* - *C shell* (%) - osnovna verzija
- *tcsh* - napredna verzija *C shella* (%)
- *ksh* - *Korn shell* ($)

Sistemske skripte: Shell
Razlike i izbor (2)

- Dobro je znati barem osnove rada u ljusci *sh* jer se prilikom instalacije i “spašavanja” sustava obično koristi
- Važna karakteristika *sh* ljuske je da je većina distribucija statički povezana pa je teže kompromitirati sustav
- Ljuska se mijenja naredbom *chsh*, i svaki korisnik na sistemu može promjeniti ulazni shell sa standardno postavljenog na željeni, ako je naredba instalirana na operacijskom sustavu (*root* može koristiti *usermod*)

Sistemske skripte: Shell

Inicijalizacija ljuske *sh*

- Konfiguracijske datoteke ljuske *sh*:

  /sbin/sh - izvršna datoteka ljuske *sh*
  /etc/profile - konfiguracijska datoteka za sve korisnike - izvršava se kod prijavljivanja na sustav
  ~/.profile - konfiguracijska datoteka za pojedinog korisnika - izvršava se kod prijavljivanja na sustav

Sistemske skripte: Shell

Inicijalizacija ljuske *bash*

- Konfiguracijske datoteke ljuske *bash*:

  /usr/local/bin/bash - izvršna datoteka ljuske *bash*
  /etc/profile - konfiguracijska datoteka za sve korisnike - izvršava se kod prijavljivanja na sustav
  ~/.bash_profile - konfiguracijska datoteka za pojedinog korisnika - izvršava se kod prijavljivanja na sustav

Sistemske skripte: Shell

Inicijalizacija ljuske *bash* (2)

~/.bashrc - konfiguracijska datoteka za pojedinog korisnika - izvršava se kod svakog otvaranja novog shella

~/.inputrc - konfiguracijska datoteka za pojedinog korisnika - sadrži informacije koje određuju način rada u naredbenoj liniji - okolinske (environment) varijable

Sistemske skripte: Shell

Inicijalizacija ljuske *bash* (3)

- Primjer *~/.bash_profile*

```
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup programs
PATH=$PATH:$HOME/bin
BASH_ENV=$HOME/.bashrc
USERNAME=""


export USERNAME BASH_ENV PATH
```

Sistemske skripte: Shell

Inicijalizacija ljuske *bash* (4)

- Primjer *~/.bashrc*

```
# .bashrc

# User specific aliases and functions

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc

fi
```

Sistemske skripte: Shell
Komunikacija

- Definicija varijabli:

```
$ TEKST="Danas je lijepi dan"
$ BROJ=13
```

- Ovako definirane varijable su lokalne
- Ako želimo već definiranu varijablu učiniti globalnom, to možemo učiniti ovako:

```
$ export TEKST
$ export BROJ
```

Sistemske skripte: Shell
Komunikacija (2)

- Ljuska komunicira s korisnikom putem:
  - *stdin* - ulazni podaci (tipkovnica)
  - *stdout* - izlazni podaci (monitor, pisač)
  - *stderr* - izlaz za ispis grešaka
- U ljusci se *stdin, stdout* i *stderr* ponašaju kao datoteke, pa ako želimo neku datoteku ispisati na ekran možemo to s

```
cp neka_datoteka.txt  /dev/stdout
```

pod uvjetom da je *stdout* usmjeren na ekran

Sistemske skripte: Shell
Komunikacija (3)

- Redirekcija (preusmjeravanje):

  spremanje rezultata izvođenja neke naredbe (*ls*) u datoteku *datoteka_ispisa*

```
$ ls -ali > datoteka_ispisa
$ ls -ali 1> datoteka_ispisa
```

Sistemske skripte: Shell
Komunikacija (4)

- Redirekcija (preusmjeravanje):

  spremanje ispisanih grešaka prilikom izvođenja neke naredbe (*ls*) u datoteku *datoteka_gresaka*

```
$ ls -ali 2> datoteka_gresaka
```

Sistemske skripte: Shell
Komunikacija (5)

- Redirekcija (preusmjeravanje):

  spremanje rezultata izvođenja neke naredbe (*ls*) i ispisanih grešaka u datoteku *ispisi*

```
$ ls -ali 1> datoteka_ispisa 2>&1
```

Sistemske skripte: Shell
Komunikacija (6)

- Redirekcija (preusmjeravanje):

  spremanje rezultata izvođenja neke naredbe (*ls*) u datoteku *ispis*, *a* ispisanih grešaka u datoteku *greske*

```
$ ls -ali 1>ispis 2>greske
```

Sistemske skripte: Shell
Komunikacija (7)

- *Pipe* (prosljeđivanje):

  kada želimo rezultat (ispis) izvršavanje jedne skripte proslijediti drugoj skripti (ili programu) možemo koristiti *Pipe* “|”. Kod izvršavanja linija :

```
$ cat /etc/passwd | grep branko
```

  će iz datoteke */etc/passwd* ispisati samo liniju koja posjeduje riječ “branko”

Sistemske skripte: Shell
Komunikacija (8)

- Vrijednost e*xit* (signalizacija uspješnosti izvođenja):

  kada želimo signalizirati ljuski ili drugoj skripti koja je pozvala skriptu da je izvođenje bilo uspješno ili neuspješno to možemo učiniti narebom *exit* koja zatvara ljusku u kojoj se izvodi skripta i prosljeđuje indikator uspješnosti.

vrijednost *exit* = 0	skripta uspješno izvedena
vrijednost *exit* > 0	nastupila je greška

Sistemske skripte: Shell

Interaktivne skripte

- Skripte koje se izvršavaju u interakciji s korisnikom (skripta se ne može izvršiti bez akcije korisnika, npr. unos određenih parametara sa tipkovnice)
- Primjer takve skripte je skripta za dodavanje novih korisnika, koja od administratora zahtijeva unošenje određenih podataka (ime i prezime korisnika, status i sl.)

Sistemske skripte: Shell

Interaktivne skripte (2)

- Primjer interaktivne skripte koja ispisuje korisnike iz zadane grupe

```
#!/bin/sh

GRUPE="/etc/group”
PASSWD="/etc/passwd”

echo "\nPostojece grupe korisnika na sustavu\n"
cat $GRUPE | cut -d":" -f1,3

echo "\nUnesi GID grupe za koju zelis ispis :”
read GR_ID
grep $GR_ID $PASSWD | cut -d":" -f1,5

echo "\nBroj korisnika na sustavu iz grupe " $GR_ID
grep -c $GR_ID $PASSWD
```

Sistemske skripte: Shell

Samostalne skripte

- Skripte koje se izvršavaju bez interakcije s korisnikom, odnosno ne zahtijevaju od korisnika upis podataka ili bilo kakvu drugu akciju s njegove strane
- Koriste se za automatiziranje poslova, npr. svakodnevno brisanje privremenih datoteka ili za rotiranje logova
- Pogodne su za pokretanje iz *Crona* (automatsko pokretanje u određeno vrijeme)

Sistemske skripte: Shell

Samostalne skripte (2)

- Primjer samostalne skripte *newsyslog*

```
#! /bin/sh
for i in `grep \^[a-z\*] /etc/syslog.conf |awk '{print $2}'|grep /var/log`
do
     test -f $i.3.Z && mv $i.3.Z $i.4.Z
     test -f $i.2.Z && mv $i.2.Z $i.3.Z
     test -f $i.1.Z && mv $i.1.Z $i.2.Z
     test -f $i.0 && (compress $i.0; mv $i.0.Z $i.1.Z)
     mv $i   $i.0
     cp /dev/null $i
     chmod 640    $i
done
#

kill -HUP `cat /etc/syslog.pid`
```

- Skripta se izvršava jednom tjedno iz *Crona* i rotira sistemske logove

Sistemski programi: C
Uvod

- Dennis Ritchie kreirao programski jezik *C* 1972. u Bell Laboratories
- Dobio je naziv "*C*" jer je bio nasljednik programskog jezika "*B*".
- *C* je bio kreiran kao programski jezik visoke razine pogodan za pisanje operacijskog sustava *UNIX*
- Danas je većina *UNIX* sustava i aplikacija pisanih na njima ostvarena u *C*-u

Sistemski programi: C
Uvod (2)

- Proširenjem *C*-a s filozofijom objektnog programiranja nastao je *C++*, trenutno najpopularniji programski jezik koji je u potpunosti kompatibilan i sa *C*-om
- Prednost *C*-a je svakako velika količina gotovih biblioteka funkcija iz svih područja informatike, matematike i ostalih znanosti
- Postoji veliki broj besplatnih *C* prevodioca, svakako najpoznatiji je *gcc*

Sistemski programi: C
Osnovne strukture

- Primjer programa pisanog u *C*-u, koji ispisuje na standardni izlaz (u našem slučaju konzola) tekst “Pozdrav svijetu”

```
#include <stdio.h>
void main ( ) {
   printf (”Pozdrav svijetu \n");
   return 0;
}
```

- #include - nalaže prevodiocu da prije prevođenja umetne izvorni kôd koji se nalazi u datoteci navedenoj između znakova < i >, u našem slučaju *stdio.h*

Sistemski programi: C

Osnovne strukture (2)

- Varijable

```
int i, j;
char c, d;
int iarray[10];
int ipointer[];
char carray[10];
char cpointer[];
```

Osnovni tipovi varijabli:

int
char
float
double

PRIMJER C PROGRAMA S VARIJABLAMA

```
main( ) {
    int a, b, c, zbroj;
    a = 1;  b = 2;  c = 3;
    zbroj = a + b + c;
    printf("Zbroj je %d", zbroj);
 }
```

Program dodjeljuje varijablama a, b i c redom brojeve 1, 2 i 3. Potom te tri varijable zbraja a rezultat sprema u varijablu zbroj. Zadnja linija kôda ispisuje na ekran poruku “Zbroj je 6”.

Sistemski programi: C
Osnovne strukture (3)

- Kontrola toka (if)

PRIMJER KÔDA S *if* NAREDBOM

```
c = getchar( );
if( c == '?' )
      printf(”Stisnuli ste upitnik ! \n");
```

OPERATORI:

== jednako je (*equal to,* .EQ. *za Fortranaše)*
!= nije jednako (*not equal to)*
> veće od (*greater than)*
< manje od (*less than)*
>= veće ili jednako (*greater than or equal to)*
<= manje ili jednako od (*less than or equal to)*

PRIMJER KÔDA S *if* NAREDBOM I LOGIČKIM OPERATOROM *ILI*

```
if ( c==’A' || c==’B' || c==’C' )
```

LOGIČKI OPERATORI U C-u:

&& I (*AND)*
|| ILI (*OR)*
! NI (*NOT)*

Sistemski programi: C
Osnovne strukture (4)

- Kontrola toka (while)

STRUKTURA *while* PETLJE

```
while (uvjet) izvršne naredbe


while (uvjet) {
    ...
    izvršne naredbe
    ...
}
```

PRIMJER PROGRAMA S *while* PETLJOM

```
main( )
 {
    char c;

    while( (c=getchar( )) != '\0' )
       putchar(c);
 }
```

Program očekuje od korisnika da na tipkovnici utipkava slova i ispisuje ih na ekran, sve dok korisnik ne utipka **break** sekvencu.

Sistemski programi: C
Osnovne strukture (5)

- Polja (arrays) i tekstualna polja (strings)

PRIMJERI C PROGRAMA S POLJIMA

```
main( )
{
   int n, c; char line[100]; n = 0;
   while( (c=getchar( )) != '\n' ) {
      if( n < 100 ) line[n] = c;
      n++;
   }
   printf("Dužina utipkanog teksta = %d\n", n);
}
```

Program očekuje od korisnika da utipkava slova na tipkovnici i sprema ih u polje. Kada korisnik stisne **Enter** tipku, program ispisuje dužinu utipkanog teksta.

Sistemski programi: C
Osnovne strukture (6)

- *For* petlja

STRUKTURA *for* PETLJE

```
for (inicijalizacija; uvjet; korak )
    izvršne naredbe;
```

MOŽE SE PISATI I OVAKO

```
inicijalizacija;
while( uvjet ) {
   izvršne naredbe;
   korak;
}
```

PRIMJER KÔDA UGNJEŽDENE (NESTED) *for* PETLJE

```
for( i=0; i<n; i++ )
    for( j=0; j<m; j++ )
       array[i][j] = 0;
```

Primjer postavlja sve elemente dvodimenzionalnog polja *array* na vrijednost 0.

Sistemski programi: C
Osnovne strukture (7)

- Funkcije i komentari

PRIMJER C KÔDA SA FUNKCIJAMA I KOMENTARIMA

```
main() {
    int hist[129];                              /* 128 legalnih karaktera + 1 */
    count(hist, 128);                           /* broji slova u hist varijabli */
    printf( ... );                              /* ovako izgledaju komentari u C-u */
 }

 count(buf, size)
    int size, buf[ ]; {
       int i, c;
          for( i=0; i<=size; i++ )
             buf[i]  =  0;                      /* postavi buf na nulu */
          while(  (c=getchar(  )) != '\0' ) {   /* čitaj do eof signala */
             if( c > size || c < 0 )
             c = size;                          /* sredi ilegalan broj slova */
             buf[c]++;
          }
    return;
 }
```

Sistemski programi: C
Prevođenje

- Kod prevođenja gotovih programa dobro je obratiti pažnju na *README* i *INSTALL* tekstualne dokumente koje se obično nalaze uključene u distribuciju programa
- *README* obično sadrži neke važne upute za korištenje programa, ili potrebne pretpostavke da bi program uopće mogao raditi. Ponekad postoje specifične *README* datoteke za pojedine operacijske sustave (*README.OS2*)

Sistemski programi: C
Prevođenje (2)

- U datoteci *INSTALL* obično se nalaze upute za samu instalaciju ili prevođenje programa, kao i eventualne preduvjete za instalaciju. Također mogu postojati upute za pojedine operacijske sustave u zasebnim datotekama
- Kod pisanja svojih programa koji idu u distribuciju širem krugu ljudi, trebalo bi obavezno uključiti u samu distribuciju *README* i *INSTALL* datoteke

Sistemski programi: C
Prevođenje (3)

- *Configure* skripta je obično skripta koja se izvršava u shellu. Dolazi s izvornim kôdom i izvršava se prije samog prevođenja programa
- Zadatak *configure* skripte je da pripremi (uskladi) parametre za prevođenje programa na navedenoj platformi
- Može biti interaktivna ili automatizirana

Sistemski programi: C
Prevođenje (4)

- Nakon što smo izvršili skriptu *configure* ona je generirala datoteku *Makefile* koja sadrži sve potrebne postavke za platformu na kojoj vršimo prevođenje.
- *Make* čita datoteku *Makefile* i prevodi program po zadanim postavkama za platformu na kojoj vršimo prevođenje
- Pokretnjem *make install* program se i automatski instalira na predviđeno mjesto

Sistemski programi: C
Prevođenje (5)

- Ako želimo prevesti program koji smo sami napisali, odnosno nemamo *configure* ni *Makefile* datoteku, onda prevođenje moramo izvršiti ručno
- Ručno prevođenje se vrši na način da pokrenemo *C* prevodilac, u našem slučaju je to besplatni *gcc*
- *Gcc* nam prevodi naš program u objektnu datoteku, koju još treba povezati s bibliotekama

Sistemski programi: C
Povezivanje

- Na *UNIX*-u standardni povezivač *(linker)* je *ld*
- *Ld* može povezati objektne datoteke koje je kreirao prevodilac kao statičke ili dinamičke
- Statički povezani programi automatski predstavljaju i izvršne programe koji ne zahtijevaju vanjske biblioteke, već su biblioteke prilikom povezivanja povezane u sam izvršni kôd, što sam kôd čini većim, ali i ne ovise o instaliranim vanjskim bibliotekama

Sistemski programi: C
Povezivanje (2)

- Dinamički povezani programi su izvršni programi koji pozivaju vanjske biblioteke, pa je sam izvršni kôd manji, ali je ovisan o vanjskim bibliotekama
- Može predstavljati problem prebacivanje programa među sličnim strojevima ako se razlikuju inačice biblioteka instaliranih na stroju, kao i eventualno podizanje biblioteka na višu inačicu

Sistemski programi: C
Ispravljanje pogrešaka

- *Cvs (Concurrent Versioning System)* je alat koji omogućuje autorima programa kontrolu izmjena na samom izvornom kôdu programa. Naročito je pogodan kod razvoja programa u koji je uključen veći broj autora.
- *Patch* je alat koji omogućuje autorima naknadnu izmjenu programa na već postojećem izvornom kôdu ili binarnim distribucijama

Sistemski programi: C
Ugađanje

- Ugađanje *(profiling)* je postupak koji omogućava mjerenje proteklog vremena kod izvođenja programa i koji usporedno na izvornome kôdu pokazuje koje linije kôda troše najviše vremena. S ovakvim podacima autor programa može ugoditi i poboljšati svoj program na način da dijelove programa koje troše najviše vremena ponovno napiše (primjeni neki brži algoritam)

Sistemski programi: C
Vježba br. 1

- U ovoj vježbi proći ćemo sve potrebne korake kod instalacije skriptnog jezika *Perl* :
  - provjera trenutne verzije na sustavu (*perl -v*)
  - brisanje stare verzije sa sustava (*dpkg -r*)
  - download nove verzije (*ftp*)
  - priprema za prevođenje (*configure*)
  - prevođenje i povezivanje (*make*)
  - instalacija (*make install)*
  - provjera nove verzije *(perl -v)*

Pomoćne skripte: Perl

Uvod

- *Perl = sed + awk + sh + Pascal + C + BASIC*
- Trenutna inačica je 5.6.1
- Uskoro treba izaći *Perl 6*
- Kombinacija interpretera i prevodioca jer prilikom pokretanja prvo prevede izvorni kôd u *Perl* meta kôd, kojeg zatim interpretira (izvršava) liniju po liniju
- Kôd se obično bez ikakvih promjena može koristiti na raznim operacijskim sustavima

Pomoćne skripte: Perl

Osnovne strukture

- Perl poznaje tri tipa verijabli: *scalar*, *array* i *hash*
- Za razliku od mnogih drugih programskih jezika, *scalari* u *Perlu* mogu primati po potrebi i numeričke i alfanumeričke vrijednosti, a *Perl* efikasno radi njihovu konverziju u zavisnosti o trenutnoj potrebi u programu

Pomoćne skripte: Perl
Osnovne strukture (2)

- *Perl* posjeduje mnoge predefinirane varijable, a najčešće korištena je standardna (default) varijabla *$_* u koju upisuje rezultate zadanih naredbi, ako mu ne kažemo drugačije

```
$days               # najjednostavniji oblik scalara imena "days”
$days[28]           # pokazuje sadržaj 29-og elementa arraya @days
$days{'Feb'}        # pokazuje sadržaj 'Feb' polja iz hasha %days
$#days              # pokazuje vrijednost zadnjeg indeksa arraya @days
```

Pomoćne skripte: Perl
Osnovne strukture (3)

- Primjer skripte pisane u *Perlu*

```
#!/usr/bin/perl

print "\nSkripta koja brise suvisne datoteke iz TUCOWS mirrora\n";
print "(c) 2001 Branko Radojevic, branko\@vdu.hr\n\n";

if (defined($ARGV[0])) {
   $dir=$ARGV[0];
   print "Provjerit cu direktorij : $dir\n";
   system ("ls -li $dir > $dir\.listing_current");
   system ("cut -b 65- $dir\.listing_current > $dir\.listing_current_files");
   system ("cut -b 57- $dir\.listing > $dir\.listing_files");

   $broj_izbrisanih = 0;
   open (CURR, $dir."\.listing_current_files");
```

Pomoćne skripte: Perl
Osnovne strukture (4)

```
while ( <CURR> ) {
    if ($_ eq "\n") { next;  }
    chomp;
    $ima=0;

open (LIST, $dir."\.listing_files");
foreach $list ( <LIST> ) {
    open (CURR, $dir."\.listing_current_files");
    while ( <CURR> ) {
              if ($_ eq "\n") { next;  }
              chomp;
          $ima=0;

              open (LIST, $dir."\.listing_files");
          foreach $list ( <LIST> ) {
              chomp $list; chop $list;
```

Pomoćne skripte: Perl
Osnovne strukture (5)

```
        if ($list eq $_)
                { $ima=1; }
        }
        if ($ima eq 1)
                { print "OK  : $_ :\n"; }
        if ($ima eq 0)
                { print "DEL : $_ :\n";
                system ("rm -r $dir$_");
                $broj_izbrisanih = $broj_izbrisanih + 1; }
        }
}


print "\nIzbrisano datoteka : $broj_izbrisanih\n";
```

Pomoćne skripte: Perl

Moduli *CPAN*

- *CPAN (Comprehensive Perl Archive Network)* je arhiva modula kojima se može nadograđivati osnovna distribucija *Perla*
- *CPAN* modulima se Perl može proširivati direktno iz *Perla*, ili iz *UNIX* ljuske
- Prilikom nadogradnje *CPAN* moduli sami detektiraju ako nedostaje neki drugi *CPAN* modul te se takvi moduli sami automatski instaliraju
- Perl može nadograditi sam sebe putem *CPAN*-a

Pomoćne skripte: Perl
Moduli *CPAN* (2)

- *CPAN* arhiva se može pretraživati po ključnim riječima direktno iz *Perla*
- Prilikom prve instalacije modula sa *CPAN*-a potrebno je navesti nekoliko parametara, a najbitniji je lokalni *CPAN* mirror.
  Preporučamo
  ftp://ftp.linux.hr/pub/CPAN/
  koji je službeni *CPAN* mirror u Hrvatskoj

Pomoćne skripte: Perl
Moduli *CPAN* (3)

- U osnovnoj distribuciji (pa i u CARNetovom službenom *Perl* paketu) dolazi određeni dio modula koji se mogu nadograđivati na novije inačice putem *CPAN*-a
- Popis svih modula koji se mogu naći na *CPAN*-u se može naći na adresi :

  http://www.cpan.org/modules/00modlist.long.html

Pomoćne skripte: Perl
Generatori kôda *SWIG*

- *SWIG (Simplified Wrapper and Interface Generator)* je generator kôda koji povezuje programe pisane u *C*-u i *C++* sa skripting jezicima kao što su *Perl, Tcl/Tk, Python, Java, Ruby, Guile* i *Eiffel*
- *SWIG* omogućuje dinamičko ili statičko povezivanje *C* i *C++* funkcija u sam *Perl* interpreter koristeći *Perl use* naredbu

Pomoćne skripte: Perl
Generatori kôda *SWIG* (2)

- *SWIG* primjer

```
/* File : example.c */

#include <time.h>
double My_variable = 3.0;

int fact(int n)  {
   if (n <= 1)
      return 1;
   else
      return n*fact(n-1); }

int my_mod(int x, int y) {
   return (x%y); }

char *get_time() {
   time_t ltime;
   time(&ltime);
   return ctime(&ltime); }
```

```
/* example.i */

%module example
%{
   /* Put header files here */
%}

extern double My_variable;
extern int fact(int n);
extern int my_mod(int x, int y);
extern char *get_time();
```

Pomoćne skripte: Perl
Generatori kôda *SWIG* (3)

- *SWIG* primjer (nastavak)

```
unix % swig -perl5 example.i
Making wrappers for Perl5
unix % gcc -c example.c example_wrap.c \
    -I/usr/lib/perl/solaris/5.003/CORE
unix % ld -G example.o example_wrap.o \
   -o example.so
unix % perl
use example;
print $example::My_variable,"\n";
print example::fact(5),"\n";
print example.get_time(),"\n";
<ctrl-d>
3.0
120
Sun Feb 11 23:01:07 1996
unix %
```

Pomoćne skripte: Perl
WWW/CGI

- *Perl* je dobar alat za izradu skripti koje kao korisničko sučelje koriste WWW
- *Perl* se može koristiti odvojeno od web poslužitelja ili može biti uključen kao modul u sam *Apache* web poslužitelj (mod_perl)
- Dobar primjer takve upotrebe je i paket *Mailman* koji služi za čitanje i pisanje E-maila putem web sučelja, a postoji i kao CARNetov službeni paket (treba zahtjev, zbog licence)

Pomoćne skripte: Perl
WWW/CGI (2)

- U samoj distribuciji *Perla* isporučuju se moduli koji služe za lakše komuniciranje između web poslužitelja i *Perla (CGI.pm)*
- Postoje i drugi moduli koji mogu biti specifični za pojedine web poslužitelje, npr. *Apache::Cookie* modul koji omogućuje jednostavno namještanje cookieja u korisnikovim web pretraživačima

Pomoćne skripte: Perl

Vježba br. 2

- U ovoj vježbi proći ćemo sve potrebne korake za pristup *CPAN-u* :
  - pokretanje *CPAN* modula unutar Perla :
    $*perl -MCPAN -e shell;*
  - kod prvog pokretanja potrebno je odgovoriti na par pitanja (s koje *CPAN* arhive će uzimati module i sl.)
  - pretražiti arhivu modula
  - instalirati izabrani modul

WWW/CGI skripte: PHP
Osnovne strukture

- *PHP* je server-side, višeplatformski, *HTML* skripting jezik, a služi za izradu dinamičkih web stranica
- Sam *PHP* predprocesor se nalazi uključen u web poslužitelj
- Web stranice koje koriste *PHP* se kreiraju kao i svake druge *HTML* stranice, a razlika je u nastavku imena, koji je obično *.php* (umjesto *.html*)

WWW/CGI skripte: PHP

Osnovne strukture (2)

- Kad web poslužitelj naiđe na web stranicu s nastavkom *.php*, prvo je propusti kroz *PHP* predprocesor, koji izvrši kôd koji se nalazi na stranici i vrati gotovu stranicu nazad web poslužitelju, koji je tek tada šalje korisniku u njegow web pretraživač
- *PHP* se trenutno nalazi na inačici 4.0
- Posjeduje module za komunikaciju s mnogim bazama podataka

WWW/CGI skripte: PHP
Osnovne strukture (3)

- Primjer *PHP* kôda (test.php)

```
<html><head><title>PHP Test</title></head>
<body>
<?php echo "Hello World<p>"; ?>
</body></html>
```

- Ako ovaj kôd snimimo u root direktorij web poslužitelja, možemo ga pozvati iz web pretraživača s

  http://www.ime.hr/test.php

WWW/CGI skripte: PHP
Osnovne strukture (4)

- Još jedan primjer *PHP* kôda

```
<?php
if(strstr($HTTP_USER_AGENT,"MSIE")) {
?>
<center><b>You are using Internet Explorer</b></center>
<?
} else {
?>
<center><b>You are not using Internet Explorer</b></center>
<?
}
?>
```

WWW/CGI skripte: PHP
Osnovne strukture (5)

- Varijable u *PHP*-u

```
$var = "Bob";
$Var = "Joe";

echo "$var, $Var";      // outputs "Bob, Joe"
$4site = 'not yet';     // invalid; starts with a number
$_4site = 'not yet';    // valid; starts with an underscore
$täyte = 'mansikka';    // valid; 'ä' is ASCII 228.
```

WWW/CGI skripte: PHP
Osnovne strukture (5)

- *If, for, do..while*

```
if ($a > $b)
    print "a je veće od b";


for ($i = 1; $i <= 10; $i++)  {
    print $i;
}


$i = 0;
do {
   print $i;
} while ($i>0);
```

WWW/CGI skripte: PHP
WWW/CGI

- *PHP* se obično nalazi uključen u distribuciju web poslužitelja, a to je u CARNetovom slučaju *Apache* web poslužitelj (*PHP* se nalazi unutar *Apache* CARNetovog službenog paketa)
- *SquirrelMail* web-mail program (koji spada u službene CARNetove web-mail programe) je u potpunosti baziran na *PHP*-u

Sažetak

- Prikazani interpeteri i prevodioci su odabir najkorisnijih programskih alata za sistem-inženjera koji mu omogućuju automatizaciju mnogih poslova vezanih za administraciju (npr. automatsko otvaranje novih korisnika na sustavu) ili kao alati za dinamičko generiranje *WWW* stranica podacima s *UNIX* sustava
- Preporučljivo je dobro vladati osnovnim *UNIX* ljuskama, te barem poznavati u osnovnim crtama programiranje u *C*-u, *Perlu* i *PHP*-u

Literatura

- http://www.shellscript.com (shell skripte)
- http://www.iu.hio.no/~mark/unix/unix.html (UNIX)
- http://www.lysator.liu.se/c/bwk-tutor.html (C)
- http://www.perl.com (Perl)
- http://www.cpan.org (CPAN)
- http://www.swig.org (Swig)
- http://www.php.net (PHP)