SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 94

PARALELIZAM U POKRETANJU OPERACIJSKIH SUSTAVA

Krunoslav Tomorad

Zagreb, lipanj 2008.

Sadržaj

1. Uvod

Svrha ovog dokumenta je osvrt na novije načine pokretanja Unix-u sličnih operacijskih sustava koji se temelje na paralelnom podizanju usluga. Time se skraćuje vrijeme potrebno za pokretanje operacijskih sustava.

Kako se kroz relativno kratko vrijeme pojavilo više programa koji iskorištavaju takve načine pokretanja, postavlja se pitanje koliko su oni učinkoviti te da li ima neke bitne razlike među njima. Glavni dio ovog dokumenta se bavi ispitivanjem najvažnijih programa iz tog područja.

Drugo poglavlje služi sa objašnjavanje načela po kojem se Unix-u slični operacijski sustavi podižu. Također, biti će objašnjen koncept paralelnog pokretanja usluga uz odgovarajuće primjere u komercijalno dostupnim operacijskim sustavima.

Nakon toga će se, u trećem poglavlju, diskutirati programi koji će se koristiti u ispitivanjima. Isto tako, detaljno će se opisati postupak instalacije svakog pojedinog programa. Nakon toga, može se pristupiti samom ispitivanju.

Ispitivanje je pokriveno četvrtim poglavljem. Opisati će se usluge koje su odabrane za ispitivanje te će biti proveden test za svaki odabrani program. Na kraju četvrtog poglavlja će se usporediti rezultati.

U dodacima se nalaze formati opisa usluga za program Initng te eINIT. Osim ključnih riječi sa pripadnim opisima, dat će se i pojašnjeni primjeri. Također, u dodacima se nalazi kratak uvod u GRUB te Bootchart, programe koji se intenzivnije koriste kroz rad.

2. *init* – Unix proces broj 1

2.1. Pregled pojmova

Da bi bilo moguće razumjeti tekst koji slijedi, potrebno je znati značenje korištenih izraza. Korišteni izrazi prikazani su tablicom 2.1.

Tablica 2.1: Popis izraza korištenih u dokumentu

Izraz	Izvorni naziv	Značenje
Spajanje diska	disk mounting	Postupak kojim se neki disk (čvrsti disk, kompaktni disk, magnetska vrpca i sl.) uklope u hijerarhiju datotečnih sustava. Nakon spajanja diska može mu se pristupati, koristiti postojeća kazala i datoteke te stvarati nova. Taj se postupak izvodi alatom `mount`.
Odspajanje diska	disk unmounting	Postupak koji provodi obrnutu funkciju od spajanja. Nakon što je disk odspojen, više se ne vidi u hijerarhiji datotečnih sustava. Postupak se izvodi korištenjem alata `umount`.
Datotečni sustav	filesystem	Logička tvorevina na mediju za pohranu podataka koja omogućava pohranu podataka na način prikladan za korištenje od strane operacijskog sustava. Podaci su najčešće pohranjeni u hijerarhiju kazala (engl. *directory*).
Točka spajanja	mount point	Mjesto (kazalo) u hijerarhiji datotečnih sustava gdje je datotečni sustav spojen. Korijen hijerarhije tog datotečnog sustava je upravo točka spajanja.
Korijenski datotečni sustav	root filesystem	Datotečni sustav koji se spaja na točku `/` u hijerarhiji datotečnih sustava. Ako ništa drugo, treba sadržavati kazala na koja će se spajati ostali datotečni sustavi.
Program	program	Binarna datoteka u kojoj se nalazi skup instrukcija koje će obavljati neki unaprijed određeni posao (npr. obrada informacija). Uz instrukcije sadrži upute operacijskom sustavu potrebne da bi se program ispravno pokrenuo. Najčešće se dobiva prevođenjem kôda pisanog u nekom višem programskom jeziku.
Proces	process	Program u izvođenju.
Identifikacijski broj procesa	PID – process identifier	Broj jedinstven za svaki proces, dodjeljuje ga operacijski sustav prilikom pokretanja programa.
Pozadinski proces, demon	daemon	Neinteraktivni proces koji se izvršava u pozadini. Ne prima ulaz od korisnika niti korisniku izravno daje povratne informacije. Namjena mu je da pruža usluge drugim procesima, a ne korisniku.

2.2. Koncept

Nakon što se operacijski sustav pokrene, očekuje se da je pokrenut čitav niz programa raznih namjena. Na primjer, zahtjeva se da je nakon pokretanja operacijskog sustava

omogućen zvuk, na *web* poslužiteljima treba biti pokrenut HTTP poslužitelj, a poslužitelji elektroničke pošte zahtijevaju da je pokrenut program za razmjenu elektroničke pošte (engl. *MTA, Mail Transfer Agent*). U krajnjoj liniji, najčešće treba biti pokrenuto neko korisničko sučelje. Pri tome, krajnjeg korisnika koji samo koristi usluge operacijskog sustava ne zanima, niti bi on trebao znati, način na koji su svi ti programi pokrenuti. Sam postupak podizanja je različit na različitim operacijskim sustavima – unutar ovog teksta će se razmatrati GNU/Linux inačica Unix operacijskog sustava.

Kako zapravo izgleda proces podizanja operacijskog sustava? Prvo se u memoriju učita jezgra operacijskog sustava – pri tome sama funkcionalnost učitane jezgre može jako varirati pošto je Linux jezgra monolitna sa podrškom za dinamičkim učitavanjem vanjskih modula. To znači da se neka funkcionalnost, primjerice upravljački program za mrežno sučelje, može prevesti (engl. *compile*) zajedno sa jezgrom i biti njezin sastavni dio (te se time učitavati zajedno s njom) ili može biti izgrađena kao zasebna cjelina (modul) koji će se kasnije učitavati po potrebi.

Nakon što je jezgra učitana, postoji mogućnost pokretanja tzv. *initrd*/*initramfs* slike. To je način na koji Linux jezgra omogućava učitavanje nekih vanjskih modula prije nego glavni datotečni sustav može biti spojen. Primjerice, ako bi se korijenski datotečni sustav (engl. *root filesystem*) nalazio na čvrstom disku koji se spaja na SATA upravljački uređaj za koji odgovarajući upravljački program nije dio same jezgre nego je u obliku vanjskog modula, tada on ne bi mogao biti spojen. No, ako se koristi *initramfs* slika, tada se u nju spremi odgovarajući modul te se sama slika podesi tako da se taj modul učita prije spajanja datotečnih sustava. Ovo je samo jednostavni primjer čija je svrha objasniti čemu *initramfs* slike služe, no potrebno je naglasiti da je korištenjem tih slika moguće napraviti i puno zahtjevnije radnje.

Konačno, nakon što jezgra iskoristi *initramfs* sliku za dio posla za koji je ona bila namijenjena, korijenski datotečni sustav je spojen te se pokreće program pod nazivom *init*. *Init* se ponekad naziva Unix proces broj 1 – to je stoga što je to prvi proces koji dobije svoje mjesto u jezgrinoj tablici procesa te mu je zbog toga identifikacijski broj procesa (engl. *PID – process identifier*) jednak 1. Program *init* se pokreće kao pozadinski proces. Svrha programa *init* je da pokrene sve ostale potrebne programe. Kako postoji više ostvarenja (engl. *implementation*) Unix operacijskog sustava, razvilo se više inačica programa *init*. Najpoznatije inačice su *SysV init* te *BSD init*. Kod konfiguracije programa *init* je potrebno

voditi računa o tome kojim redoslijedom se podižu programi jer oni često ovise jedni o drugima. Tako se, na primjer, ne smije pokrenuti *web* poslužitelj prije nego su podešena mrežna sučelja. U slučaju da se to dogodi, *web* poslužitelj se ne bi uspio pokrenuti jer bez podešene mreže ne može raditi, u nekom trenutku iza toga bi se mreža podesila no *web* poslužitelj će ostati nepokrenut. Program *init* je također zadužen za gašenje računala – prilikom toga treba pogasiti sve procese te na kraju i ugasiti samo računalo.

2.3. Tradicionalna ostvarenja programa *init*

Kao što je već ranije spomenuto, dvije najpoznatije tradicionalne izvedbe programa *init* su *SysV init* te *BSD init*. Razlika između njih je način na koji rade – dok *BSD init* izvršava jednu skriptu, *SysV init* uvodi koncept tzv. razina pokretanja (engl. *runlevel*). Obje izvedbe imaju svoje prednosti i mane.

2.3.1. BSD init

Program *BSD init* [7] prilikom pokretanja izvršava skriptu koja se nalazi u kazalu `/etc/rc/`. To je skripta ljuske (engl. *shell script*) iz koja podiže sve ostale programe. To je jednostavan način pokretanja operacijskog sustava koji ne unosi preveliki pretek (engl. *overhead*), tj. zahtjeva relativno malo vremena za vlastitu inicijalizaciju. S druge strane koncept jedne skripte za cijeli sustav je prilično rizičan jer nemogućnost pokretanja jednog programa može onemogućiti pokretanje cijelog sustava. Usto, teže je napraviti program koji bi automatski i pouzdano, bez intervencije krajnjeg korisnika, dodao dio skripte za pokretanje nekog novog programa, na primjer prilikom njegove instalacije, u odgovarajući dio glavne skripte.

2.3.2. SysV init

Program *SysV init* [7] pruža mnogo robustniju, ali i zamršeniju infrastrukturu za podizanje programa. Sve se vrti oko koncepta razina pokretanja [8]. Postoji osam razina pokretanja koje se označavaju brojevima *0-6* te slovom *S* (ili *s*). Od toga su tri razine rezervirane, a ostale se mogu koristiti po želji (no u određenom operacijskom sustavu je određeno za što se koja razina koristi). Popis rezerviranih razina i njihovih opisa je dan u tablici 2.2. *SysV init* nakon pokretanja provjerava koja je razina pretpostavljena (to je zapisano uz ključnu riječ "initdefault" u datoteci `/etc/inittab`) te zatim ulazi u tu razinu. Ako ne postoji pretpostavljena razina, od korisnika se traži da ručno unese razinu u koju će sustav ući.

Jednom kada je sustav pokrenut, trenutna razina se može provjeriti naredbama `runlevel` (nije dostupna svuda) te `who`:

```
$ runlevel
N 2
$ who -r
      run-level 3  Jun  7 17:05             last=#
```

Tablica 2.2. Popis rezerviranih razina pokretanja kod programa SysV init

Broj razine pokretanja	Opis
0	Razina koja služi za gašenje računala (engl. *halt*)
1	Jednokorisnička razina, samo se korisnik *root* može prijaviti na sustav
6	Služi za ponovno pokretanje računala (engl. *reboot*)

Svaka razina ima svoje kazalo u kojem se nalaze skripte (ili češće simboličke poveznice, engl. *symbolic link*) koje se pokreću prilikom ulaska te izlaska iz te razine. U slučaju simboličkih poveznica, iste pokazuju na odgovarajuću skriptu unutar kazala `/etc/init.d/`. Same poveznice su smještene u unutar `/etc/` kazala: `/etc/rcX.d/`, gdje `X` predstavlja brojku/slovo te razine pokretanja. Na nekim GNU/Linux distribucijama postoji i shema u kojoj se nalazi kazalo `/etc/init/` unutar kojeg se dalje nalaze prije navedena kazala. Razina *S* se pokreće jednom prilikom podizanja sustava (te se izvršava čak i kada se sustav pokreće u jednokorisničkoj razini). U toj se razini nalaze skripte koje podižu programe nužne za normalan rad sustava, primjerice spajanje datotečnih sustava osim korijenskog, postavljanje imena računala (engl. *hostname*) te sinkroniziranje sustavskog sata sa sklopovskim.

Sa stajališta usluga koje pružaju, skripte se često zovu usluge (servisi, engl. *services*). U daljnjem tekstu će se zbog veće razumljivosti te konzistentnosti koristiti izraz skripta, iako svako takvo mjesto može slobodno doći i riječ poveznica. Iz ispisa dolje se može vidjeti primjer jedne tipične simboličke poveznice kod programa *SysV init*:

```
$ ls -l /etc/rc2.d/S20ssh
lrwxrwxrwx 1 root root 13 Apr 16 12:55 /etc/rc2.d/S20ssh ->
../init.d/ssh
```

Iz ispisa je vidljivo da ime poveznice nije jednostavno `ssh` već `S20ssh`. To je zbog nomenklature programa *SysV init*. Već je rečeno da se skripte izvršavaju prilikom ulaska i izlaska iz pojedine razine. Pošto je postupak gašenja programa različit od njegovog

pokretanja, tako se niti skripte ne mogu u oba slučaja na isti način pozivati. Taj je problem riješen tako da se prilikom ulaska u neku razinu pokreću samo skripte koje počinju slovom S (od engl. *start*), a prilikom izlaska iz njega skripte koje počinju slovom K (od engl. *kill*) i to redoslijedom koji je određen sa dvoznamenkastim brojem koji slijedi iza slova S, odnosno K. Kada se odabere odgovarajuće ime za neku *S-skriptu*, ime za ekvivalentnu *K-skriptu* se tvori tako da se stavi slovo K te iza njega dvoznamenkasti broj dobiven oduzimanjem broja uz *S-skriptu* od broja 100, te na kraju ime usluge. To mora biti tako zbog međuovisnosti između usluga koje skripte pružaju – ako program A ovisi o programu B tada se program B mora pokrenuti prije programa A, ali se mora ugasiti nakon gašenja programa A.

Kako su i *S-skripte* te *K-skripte* najčešće poveznice na skripte unutar `/etc/init.d/` kazala i to tako da komplementarne *S-* i *K-skripte* pokazuju na istu skriptu, postavlja se novo pitanje – kako skripta zna da treba neki program pokrenuti odnosno ugasiti? Postoje dva načina kako bi se to moglo riješiti. Prvi bi zahtijevao od programera koji piše skriptu da vodi računa o tome na način da provjeri ime skripte koja se izvodi (provjerom varijable `$0`). Naime, iako skripta ima svoje ime (primjerice `ssh`), ako ju pozovemo preko neke poveznice vrijednost varijable `$0` će biti naziv te poveznice, a ne izvorne skripte. Drugi način na koji se to može riješiti je da sam program *init* brine o tome – tako radi *SysV init*. Ukoliko se neka razina pokreće, izvršavaju se abecednim redoslijedom samo skripte koje počinju slovom S, ali tako da im se kao prvi argument naredbenog retka (`$1`) prenese niz "start". Ekvivalentno, skriptama koje trebaju ugasiti neki proces se prenosi niz "stop". Od programera se zahtjeva da prilikom pisanja skripte provjeri koji argument je prenesen te da se na odgovarajući način dalje izvršava skripta. To se tipično radi tako da se napišu funkcije `start()` i `stop()` koje rade glavni dio posla, te glavni dio skripte koji provjerava parametre te poziva odgovarajuću funkciju. Skripte često primaju i druge parametre osim "start" i "stop", no njih u načelu ne koristi program *init* već su tu da pomažu krajnjim korisnicima. Primjeri takvih dodatnih parametara su: "restart" koji se često ostvaruje tako da se najprije pozove funkcija `stop()`, a odmah iza toga `start()` te "status" koji na standardni izlaz ispisuje da li je usluga pokrenuta ili ne (i vraća odgovarajuću povratnu vrijednost što omogućava korištenje unutar drugih skripti/programa). Primjer ručnog korištenja skripti će biti pokazan za usluzi zaduženoj za ispisivanje:

```
$ /etc/init.d/cupsys status
Status of Common Unix Printing System: cupsd is not running.
```

Na nekim GNU/Linux distribucijama postoje alati naredbenog retka za upravljanje uslugama, npr. na distribucijama kojima je osnova RedHat Linux postoji alat (izvedbeno je riječ o skripti ljuske) `service`. Primjer upotrebe tog alata na gornjoj usluzi je dan ispod:

```
$ /sbin/service cupsys status
Status of Common Unix Printing System: cupsd is not running.
```

Program *SysV init* je skalabilan i robustan te je unutar njega lako automatizirano dodavanje usluga. Postavljanje neke usluge tako da se pokreće automatski u nekoj razini se svodi na jednostavno stvaranje simboličke poveznice. Međutim, to se plaća većom složenošću koja dovodi do većeg preteka – *SysV init* zahtjeva više procesorskog vremena za vlastite potrebe (tzv. kućanske poslove) od programa *BSD init*.

Bez obzira koje ostvarenje programa *init* se koristi, uz program *init* tipično dolazi i nekoliko alata, poput:

- `shutdown` – omogućava gašenje te ponovno pokretanje računala s mogućnošću vremenske odgode te obavještavanju korisnika o tome,
- `halt` – gasi računalo, na *SysV initu* poziva razinu 0,
- `reboot` – ponovno pokreće računalo, na *SysV initu* poziva razinu 6,
- `telinit` – omogućava promjenu trenutne razine.

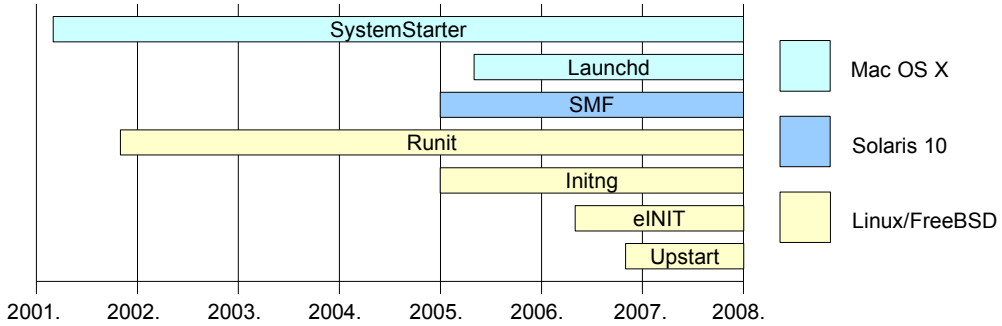
2.4. Koncept paralelnog pokretanja usluga

Iako je tradicionalni model podizanja Unixu sličnih operacijskih sustava veoma konfigurabilan i skalabilan, s vremenom je postalo izvjesno da je takav model zastarjeli. Kako je sklopovlje postajalo sve brže, to je pogodovalo razvijanju programske podrške pogodne za korištenje od strane krajnjih korisnika. Nastao je cijeli niz novih programskih rješenja koja su se morala pokretati prilikom podizanja sustava. To je nužno dovelo do sporijeg podizanja sustava. Dok je to još donekle u redu na poslužiteljima (koji se u načelu rijetko podižu pošto znaju ostati pokrenuti i po nekoliko mjeseci), na osobnim računalima (čiji je broj daleko veći od broja poslužitelja) je to posve neprihvatljivo.

Ako se malo bolje pogleda postupak podizanja operacijskog sustava, može se primijetiti da procesor dosta vremena besposličari. Najbolji primjer za ilustraciju toga su mrežne usluge. Na primjer, podiže se neka mrežna usluga i prilikom toga treba komunicirati sa nekim poslužiteljem. No, poslužitelj je zauzet drugim zahtjevima pa je spor ili je čak i nedostupan. Usluga koja se podiže tipično ima neko vrijeme koje čeka na odgovor od poslužitelja (engl.

timeout) – za to vrijeme procesor ne radi ništa korisno. Usluga koja je sljedeća na redu za podizanje se neće početi podizati sve dok se usluga koja se trenutno podiže ne podigne do kraja.

Na osnovu takve analize se može doći do ideje paralelnog podizanja usluga. Zajedno s podizanjem maloprije spomenute mrežne usluge bi se mogla pokretati neka druga usluga koja ne ovisi o toj mrežnoj usluzi. To je osnovni koncept koji se iskorištava kod modernih ostvarenja programa *init*. Programi *init* sa paralelnim podizanjem usluga se standardno isporučuju sa mnogim modernim komercijalnim, Unixu sličnim, operacijskim sustavima, npr. na Mac OS X (launchd) te na Sun Solaris-u 10 (SMF). Isto tako, postoje slični programi i za GNU/Linux i FreeBSD operacijske sustave. Vremenska crta nastanka nekih ostvarenja dana je na slici 2.1.



Slika 2.1. Vremenska crta nekih modernih ostvarenja programa init

2.4.1. launchd

Kada je tvrtka Apple Computer 2000. godine izdala novu inačicu svojeg operacijskog sustava (Mac OS X), to je bio veliki preokret jer su tim činom prešli na Unix osnovu. Sa svakom novom inačicom, tvrtka Apple je dodavala nova tehnološka rješenja u svoj operacijski sustav – kako rješenja u korisničkom sučelju, tako i rješenja koja se ne vide. Inačice Mac OS-a X prije v10.4 su za pokretanje sustava koristile program SystemStarter koji je na jednostavan način vodio računa o međuovisnostima među uslugama. Opis svake usluge je sadržavao ključne riječi "Uses", "Requires" i "Provides", iza kojih se nalazio popis pseudo-usluga koje dotična usluga koristi, zahtjeva ili pruža, respektivno. SystemStarter je na osnovu tih opisa određenim redoslijedom podizao usluge. S pojavom Mac OS-a X v10.4, SystemStarter je zamijenjen programom launchd [11] [12]. Za razliku od SystemStartera koji je zamišljen samo kao zamjena za program *init*, launchd je zamišljen kao zamjena za

čitav niz programa. Svrha toga je smanjenje vremena potrebnog za podizanja sustava na sporijim računalima. Launchd se sastoji od dva dijela: `launchd` te `launchctl`.

`launchd` je pozadinski proces koji je, slično *init*-u, prvi proces koji se podiže. On dalje pokreće sve ostale procese. Postoje dva kazala koja pretražuje u potrazi za tzv. p-listama: `/System/Library/LaunchAgents/` te `/Library/LaunchDaemons/`. Svaka p-lista sadrži podatke o programu koji će se pokrenuti. Razlog zbog kojeg launchd pokreće procese brže od SystemStarter-a leži u činjenici da launchd zapravo ne podiže uvijek sve usluge. To se odnosi na pozadinske procese podešene da se ne pokreću automatski (pomoću ključne riječi "OnDemand" u pripadnoj p-listi). Naime, kad launchd dođe do takve usluge, umjesto da ju pokrene, rezervirati će joj potrebne pristupe (engl. *port*) te umjesto te usluge će slušati na tom pristupu. Kada se pojavi zahtjev za tom uslugom na nekom od pristupa koje je usluga registrirala, launchd će ju pokrenuti te će se sve dalje odvijati kao da je ta usluga bila pokrenuta cijelo vrijeme. Takav način rada veoma podsjeća na tradicionalni Unix program *inetd*. Štoviše, launchd je zamijenio program *inetd* na Mac OS-u X.

`launchctl` je alat naredbenog retka kojim se na jednostavan način može manipulirati uslugama, npr. pomoću njega se može pokrenuti ili zaustaviti neka usluga. Po funkciji je sličan alatu `services` koji se isporučuje sa distribucijama kojima je osnova RedHat Linux.

Izvorni kod programa launchd je bio javno dostupan od samog početka no licenca pod kojom je bio izdan ga nije svrstavala pod slobodne programe (engl. *free software*). Nakon nekog vremena, tvrtka Apple je popustila zahtjevima korisnika te je promijenila licencu u GNU GPL kompatibilnu (Apache Licence v2.0) što ga čini slobodnim programom.

2.4.2. Service Management Facility – SMF

SMF (skraćeno od engl. *Service Management Facility*) [9] [10] je program koji je odgovoran za podizanje Solaris 10 operacijskog sustava. Za razliku od Launchd-a, SMF je zamjena samo za program *init*. Prije inačice 10, na Solarisu se koristio klasični SysV *init*.

SMF usluge se mogu podesiti tako da se pokreću sa ograničenim ovlastima, za razliku od tradicionalnog programa *init* gdje se sve usluge pokreću sa ovlastima korisnika *root*. Time se povećava sigurnost sustava jer je, ukoliko bi usluga počela raditi nepoželjne stvari, šteta nastala time manja nego da je usluga pokrenuta sa ovlastima korisnika *root*. SMF sve svoje usluge tretira kao "objekte prvog razreda", što znači da su one više od programa kojeg

izvršava operacijski sustav. Usluge mogu imati posebna stanja koja omogućavaju njihov nadzor i kontrolu nad njima.
SMF je, kao i OpenSolaris, licenciran pod CDDL-om (engl. *Common Development and Distribution License*), čime je on slobodan program, iako nije GPL kompatibilan.

2.4.3. Ostala ostvarenja

Tradicionalno, za Linux se s vremenom pojavio čitav niz raznih ostvarenja programa *init* sa paralelnim pokretanjem usluga. O njima neće biti previše riječi u ovome poglavlju pošto su važnija ostvarenja detaljno obrađena u narednom poglavlju.
Upstart je ostvarenje koju je razvila tvrtka Canonical za potrebe distribucije Ubuntu (koju također razvijaju u toj tvrtci), no može se koristiti i u drugim distribucijama pošto je Upstart slobodan program (licenciran pod GNU GPL licencom). Za razliku od većine drugih ostvarenja, cilj Upstart-a je zadržavanje kompatibilnosti sa programom *SysV init*, ali pritom iskoristiti koncept paralelnog podizanja usluga. Postoji namjera razvojnog tima da Upstart s vremenom zamjeni i neke druge tradicionalne programe, slično Launchdu, no u trenutku pisanja ovog dokumenta to još uvijek nije postignuto.
Initng je ostvarenje čija je osnova tzv. *i*-format konfiguracije pojedine usluge, iako novije inačice podržavaju XML format. Osim podataka o tipu i načinu pokretanja usluge, *i*-format sadrži dijelove koje izvršava ljuska – ti dijelovi su ugrađene skripte ljuske. Initng je slobodan program, licenciran je pod GNU GPL licencom.
eINIT je ostvarenje koje se temelji na XML formatu datoteka, većina usluga je predstavljena posebnom XML datotekom. Ta datoteka ne sadrži nikakve dijelove koje bi izvršavala ljuska, postoji samo popis naredbi koje će eINIT izravno izvršiti prilikom podizanja te usluge. eINIT je licenciran pod 3-klauzulnom BSD licencom, čime je eINIT slobodan program.
Runit je još jedno ostvarenje, specifično po veoma maloj veličini. Koristi se najviše u ugradbenim računalima, iako ga se uspješno može koristiti i na osobnim računalima. Dio je Busybox programa. Runit je licenciran pod 3-klauzulnom BSD licencom.

3. Odabir ostvarenja za ispitivanje

Za operacijski sustav Linux se s vremenom pojavio čitav niz raznih ostvarenja programa *init*. Pri tome, nemaju sva ostvarenja istu namjenu. Primjerice, Runit je ostvarenje namijenjeno ponajprije ugradbenim sustavima. U ovom radu će se ispitivati samo nekoliko najvažnijih ostvarenja. Sljedeća ostvarenja su odabrana:

- *SysV init* [14] – samo radi usporedbe,
- Upstart [5] – zbog široke prihvaćenosti od strane poznatih distribucija,
- Initng [1] te
- eINIT [3] – zbog popularnosti.

Svi testovi će biti izvedeni na distribuciji Debian GNU/Linux 4.0. Osim osnovne instalacije, zbog potrebe izgradnje nekih ostvarenja iz izvornog kôda (engl. *source code*) biti će potrebno dodatno instalirati sljedeće programe: gcc, GNU Autotools te programska zaglavlja za glavnu C knjižnicu (libc) i Linux jezgru. Također, u ispitivanjima su se koristili neki programi koji se ne instaliraju po pretpostavci. Spomenuti programi i zaglavlja su upakirani u pakete koji se mogu instalirati na sljedeći način:

```
# apt-get remove exim4 exim4-base
# apt-get install gcc automake autoconf libc6-dev linux-
headers-2.6.18-5 free-java-sdk ant postfix apache2 slapd mysql
xinetd
```

3.0.1. SysV Init

Iako ostvarenja programa *init* sa paralelnim pokretanjem procesa pružaju mnoge prednosti u usporedbi sa klasičnim programima *init*, većina GNU/Linux distribucija se još uvijek isporučuje sa *SysV initom*. Iznimke su distribucije Ubuntu, koja još od inačice 6.06 koristi Upstart, Fedora koji u inačici 9 koristi također Upstart te još neke manje poznate distribucije koje koriste neke od ostvarenja.

SysV init će se pojaviti u ispitivanjima samo da bi se mogla povući usporedba između njega i modernijih ostvarenja. Pošto je na distribuciji Debian GNU/Linux *SysV init* pretpostavljen, nikakva posebna instalacija nije potrebna.

Potrebno je samo, radi ispitivanja, napraviti novi unos u konfiguracijskoj datoteci programa zaduženog za pokretanje operacijskih sustava (engl. *bootloader*). Pošto postoji više takvih programa, ovdje će se razmatrati program GRUB koji se koristi na gotovo svim modernim GNU/Linux distribucijama, uključujući Debian. Kratak uvod u program GRUB dan je u dodatku C.

U datoteku `/boot/grub/menu.lst` potrebno je dodati sljedeći odsječak:

```
title      Debian GNU/Linux - SysV init
root       (hd0,0)
kernel     /boot/vmlinuz-2.6.18-5-k7 root/dev/hda1 ro \
init=/sbin/bootchartd
initrd     /boot/initrd.img-2.6.18-5-k7
savedefault
boot
```

3.1. Upstart

3.1.1. Povijest

Upstart je projekt tvrtke Canonical iz Velike Britanije. Ta je tvrtka najviše poznata po svojoj distribuciji Ubuntu. Cilj Upstarta je napraviti ostvarenje programa *init* sa paralelnim podizanjem usluga, ali pritom zadržavajući kompatibilnost sa klasičnim *SysV initom*. Pod time se misli da je zadržan koncept razina pokretanja – skripte uslugâ se nalaze na istom mjestu (kazalo `/etc/init.d/`) te sa Upstartom dolaze isti alati kao i sa *SysV initom* (poput `shutdown`, `halt`, `reboot`, `telinit`, `poweroff`). Dodatno, uz klasične *SysV init* skripte, Upstart podržava i svoj vlastiti oblik usluga u obliku datoteka, koje se tada zovu poslovi (engl. *job*).

3.1.2. Načelo rada

Ako se koristi u načinu rada sa zadržanom kompatibilnošću, krajnji korisnik vidi razliku između Upstarta i *SysV inita* samo kao posebne komentare u skriptama usluga. Ti posebni komentari su zapravo ključne riječi koje služe za organiziranje međuovisnosti među uslugama. Time što su ključne riječi zakomentirane se osigurava kompatibilnost sa ljuskom – ljuska će ključne riječi zanemariti, ali gramatički analizator (engl. *parser*) programa Upstart će ih prepoznati te izgraditi popis međuovisnosti. U nastavku slijedi primjer dijela skripte gdje se nalaze podaci o međuovisnostima. Primjer je preuzet iz skripte zadužene za pokretanje Bluetooth pozadinskog procesa.

```
### BEGIN INIT INFO
# Provides:          bluetooth
# Required-Start:    $local_fs $syslog $remote_fs
# Required-Stop:     $local_fs $syslog $remote_fs
# Should-Start:
# Should-Stop:
# Default-Start:     2 3 4 5
```

```
# Default-Stop:       0 1 6
# Short-Description:  Start bluetooth daemons
# Description:
### END INIT INFO
```

Iz ispisa se vidi da blok počinje ključnom riječi `### BEGIN INIT INFO`, iza čega slijedi glavni dio te konačno ključna riječ za zaključenje bloka (`### END INIT INFO`). Popis ključnih riječi s pripadnim objašnjenjem je dan u tablici 3.1.

Tablica 3.1. Popis ključnih riječi usluga programa Upstart

Ključna riječ	Opis ključne riječi
Provides	Označava koju usluga skripta pruža.
Required-Start	Označava usluge koje moraju biti pokrenute prije nego se pokreće usluga koju pruža promatrana skripta.
Required-Stop	Označava usluge koje ne smiju biti pokrenute u trenutku pokretanja usluge koju pruža promatrana skripta.
Should-Start	Usluge koje se nalaze iza ove ključne riječi ili moraju biti pokrenute prije nego se pokreće usluga koju pruža promatrana skripta ili moraju biti u postupku podizanja.
Should-Stop	Usluge koje se nalaze iza ove ključne riječi ili moraju biti zaustavljene u trenutku zaustavljanja promatrane usluge ili moraju biti u postupku zaustavljanja.
Default-Start	Iza ove ključne riječi slijedi popis razina podizanja. Prilikom ulaska u navedene razine, podizati će se promatrana usluga.
Default-Stop	Iza ove ključne riječi slijedi popis razina podizanja. Prilikom ulaska u neku od navedenih razina, promatrana usluga će se zaustaviti.
Short-Description	Iza ove ključne riječi slijedi kratak opis usluge.
Description	Iza ove ključne riječi slijedi opširniji opis usluge.

Kao što je već spomenuto, uz *SysV init* skripte, Upstart ima podršku i za vlastiti oblik usluga – poslove. Te usluge se nalaze u kazalu `/etc/event.d/`. Detaljan opis oblika tih poslova je dan u dodatku A.

3.1.3. Instalacija

Upstart koristi *GNU Autotools* za izgradnju što znači da je, pored prevodioca, potrebno imati instalirane sljedeće programe: GNU Autoconf, GNU Automake te GNU Libtool. Rezultat izgradnje je nekoliko izvršnih datoteka i knjižnica (engl. *library*). Popis sa pripadnim opisom je dan u tablici 3.2.

Tablica 3.2. Popis alata i knjižnica dobivenih izgradnjom programa Upstart

Naziv programa/knjižnice	Opis
init	Sam program *init*.
initctl	Alat koji omogućava upravljanje uslugama za vrijeme dok je program *init* pokrenut.
logd	Pozadinski proces koji zapisuje u dnevnik (engl. *log file*) izlaz koji proizvode usluge, ako ne postoji odgovarajući terminal.
libupstart.0.0.0	Glavna knjižnica programa Upstart.
libnih.so.0.0.0	O ovoj knjižnici ovisi Upstart.

Sljedeći koraci opisuju postupak izgradnje programa Upstart:

1. Sa stranice projekta [5] se skine Upstart. Datoteka se može skinuti primjerice u kazalo `/usr/src/`:

```
$ cd /usr/src/
$ wget
http://upstart.ubuntu.com/download/0.3/upstart-0.3.9.tar.b
z2
```

2. Raspakira se arhiva programa Upstart:

```
$ tar xf upstart-0.3.9.tar.bz2
```

3. Izvrši se `configure` skripta – parametar `--enable-threading` omogućava višedretvenost krajnjih programa, `--enable-static=no` označava da se neće izgraditi statičke knjižnice, a `--enable-compat=sysv` omogućava zadržavanje kompatibilnosti sa programom *SysV init*. Argument --`prefix` služi za određivanje početnog kazala hijerarhije u koju će se instalirati Upstart. U ovom slučaju, koristiti će se kazalo `/opt/upstart/`. Taj argument je potrebno poslati skripti stoga što se, zbog kompatibilnosti sa *SysV init*-om, krajnji program zove `init` (a ne npr. `upstart`) te bi se, ako bi se kao predmetak stavilo uobičajeno kazalo `/`, prebrisao originalni *SysV init* koji dolazi sa distribucijom Debian GNU/Linux. To nije poželjno jer je i on potreban za ispitivanje.

```
$ # kazalo mora biti stvoreno prije pozivanja skripte
# mkdir /opt/upstart/
$ cd upstart-0.3.9/
$ ./configure --prefix=/opt/upstart --enable-threading \
--enable-static=no --enable-compat=sysv
```

4. Nakon što je korak 3 gotov, odgovarajuća `Makefile` datoteka je stvorena te se može pozvati program `make`:

```
$ make
```

5. Na kraju je potrebno još samo instalirati izgrađene alate i knjižnice. Za to su nam potrebne ovlasti korisnika *root*:

```
# make install
```

Osim programa Upstart, potrebno je još raspakirati pripadne poslove u odgovarajuće kazalo. Sljedeći koraci opisuju taj postupak:

1. Sa stranice projekta [5] skine se arhiva sa poslovima. Opet se koristi kazalo `/usr/src/`:

```
$ cd /usr/src/
$ wget http://upstart.ubuntu.com/download/example-
jobs/0.3/example-jobs-0.3.9.tar.gz
```

2. Odabiremo kazalo `/usr/local/etc/event.d/` iz razloga jer su i ostale datoteke instalirane u hijerarhiji `/usr/local/`. Za ovaj korak su nam također potrebne ovlasti korisnika *root*.

```
$ cd /usr/local/etc/event.d/    # ovo kazalo je stvorio
$                           # Upstart prilikom instalacije
# tar xf /usr/src/example-jobs-0.3.9.tar.gz
```

3. Nije nužno, no mogu se obrisati datoteke sa izvornim kodom ukoliko su nepotrebne:

```
$ cd /usr/src/
# rm -r upstart* example-jobs*
```

3.1.4. Podešavanje

Jednom kada je Upstart instaliran, da bi ga sustav koristio kao glavni program *init*, potrebno je, kao i kod *SysV inita*, odgovarajuće podesiti konfiguracijsku datoteku programa GRUB. Da bi se operacijski sustav pokrenuo sa programom Upstart kao prvim procesom, potrebno je dodati sljedeći odsječak u datoteku `/boot/grub/menu.lst`:

```
title      Debian GNU/Linux - Upstart
root       (hd0,0)
kernel     /boot/vmlinuz-2.6.18-5-k7 root/dev/hda1 ro \
init=/sbin/bootchartd_upstart
initrd     /boot/initrd.img-2.6.18-5-k7
savedefault
boot
```

Preko parametra "init" prenosi se staza do Bootchart skripte, ne izravno do programa Upstart. To je stoga što će program Bootchart biti korišten kasnije kod ispitivanja. Za uobičajeno korištenje (bez ispitivanja) bi se preko tog parametra prenijela staza do pravog

programa Upstart (konkretno `/opt/upstart/sbin/init`). Više informacija o programu Bootchart može se pronaći u dodatku D.
Da bi se sustav pokrenuo sa programom Upstart, potrebno je odabrati stavku "Debian GNU/ Linux – Upstart" u izborniku programa GRUB.

3.2. Initng

3.2.1. Povijest

Projekt Initng je započeo Jimmy Wennlund 2005. godine. U međuvremenu se razvojni tim promijenio te ga sad radi grupa ljudi iz cijelog svijeta. Projekt se aktivno razvija, u trenutku pisanja ovog dokumenta aktualna inačica programa Initng je bila 0.6.10.2, dok je aktualna inačica paketa Initng usluga bila 0.1.5. Dosad je većina potrebnih usluga napravljena i uključena u glavni paket usluga. Ostatak usluga koji još nije uključen u glavni paket najvjerojatnije postoji na Initng forumu [16] te se može slobodno skinuti. Razvojni tim trenutno radi na inačici 0.7.0. Uz izvorni kôd, postoje paketi za više različitih distribucija.

3.2.2. Načelo rada

Konfiguracijske datoteke kao i same usluge se nalaze u kazalu `/etc/initng/`. Iako se usluge mogu nalaziti odmah u tom kazalu, preporuka je, radi preglednosti, stavljati ih u podkazala. Za tu svrhu postoji više podkazala: `daemon/`, `net/`, `service/` te `system/`. U kazalo `daemon/` se stavljaju usluge koje se podižu kao pozadinski procesi, u kazalu `system/` se nalaze usluge koje obavljaju inicijalizaciju raznih sustavskih usluga (na pr. sinkronizacija sata, spajanje datotečnih sustava i sl.) dok se u kazalu `net/` nalaze usluge vezane uz postavljanje mrežnih sučelja. Sve ostale usluge se smještaju u kazalo `service/`.
Postoji još i kazalo `runlevel/` u kojem se nalaze konfiguracije razina pokretanja. Pretpostavljena razina je naziva `default.runlevel`. U nastavku je primjer datoteke jedne jednostavne razine pokretanja.

```
daemon/acpid
daemon/syslogd
net/all
runlevel/system
service/alsasound
service/alsasound/cards
service/alsasound/mixerstate
daemon/slim
```

Prva dva retka pokreću pozadinske procese za upravljanje potrošnjom energije te dnevnik sustava. Nakon toga se pokreće usluga `net/all` koja je izvedbeno smještena u datoteci `net/net.i`, a podešava sva mrežna sučelja na sustavu. Iza toga se može vidjeti da se podiže još jedna razina pokretanja: `runlevel/system` – to je virtualna razina koja podiže najbitnije usluge iz kazala `system/`, poput postavljanja imena računala, sinkronizacija sata, spajanje datotečnih sustava i sl. Ta virtualna razina je po izgledu potpuno jednaka pretpostavljenoj razini. Na kraju, podešava se poslužitelj zvuka te se podiže program za grafičku prijavu korisnika na sustav.

3.2.3. Instalacija

Initng za svoju izgradnju koristi program CMake – za ovu inačicu Initng-a (0.6.10.2) potrebna je inačica CMake 2.2.0 ili novija. Dodatno, potrebna je knjižnica ncurses sa pripadnim zaglavljima – ako postoji, ugradit će se podrška za ispis poruka u boji.
Sljedeći koraci opisuju izgradnju programa Initng:

1. Sa stranica projekta [1] se skine izvorni kôd programa Initng. Pritom se kao kazalo za skidanje može koristiti `/usr/src/`:

```
$ cd /usr/src/
$ wget
http://download.initng.org/initng/v0.6/initng-0.6.10.2.tar
.bz2
```

2. Raspakira se arhiva programa Initng:

```
$ tar xf initng-0.6.10.2.tar.bz2
```

3. Promijeni se radno kazalo tamo gdje je raspakirana arhiva. Razvojni tim preporučuje da se izgradnja pokrene iz novog kazala jer bi u suprotnom moglo doći do neželjenih problema. CMake će stvoriti datoteku `Makefile` koja je potrebna za sljedeći korak.

```
$ cd initng-0.6.10.2/
$ mkdir build
$ cd build/
# cmake ..
```

4. Pokreće se izgradnja programa:

```
$ make
```

5. Na kraju je potrebno još instalirati program. Za to su nam potrebne ovlasti korisnika *root*:

```
# make install
```

Osim programa Initng potrebno je izgraditi skup pripadnih usluga. Pošto je postupak potpuno identičan izgradnji programa Initng, postupak prikazan ovdje je skraćen i bez suvišnih opisa. Ispisi pozivanih programa su maknuti radi preglednosti.

```
$ cd /usr/src/
$ wget http://download.initng.org/initng-ifiles/v0.1/initng-
ifiles-0.1.5.tar.bz2
$ tar xf initng-ifiles-0.1.5.tar.bz2
$ cd initng-ifiles-0.1.5/
$ mkdir build/
$ cd build/
$ cmake ..
$ make
# make install
```

Eventualno se na kraju mogu obrisati suvišne datoteke sa diska:

```
$ cd /usr/src/
$ rm -rf initng*
```

3.2.4. Podešavanje

Nakon što je Initng sa pripadnim uslugama instaliran, potrebno je odgovarajuće podesiti razine pokretanja. To se može napraviti na dva načina. Prvi je ručno uređivanje datotekâ sa opisima razinâ, no postoji i drugi, jednostavniji, način. Naime, sa programom Initng je došao alat `genrunlevel` koji omogućava stvaranje odgovarajućih razina pokretanja na osnovu trenutno instaliranih programa. Postupak je sljedeći:

```
# genrunlevel -all
```

Istim alatom se mogu migrirati usluge ako je prije nove inačice Initnga bila instalirana starija.

Potrebno je odgovarajuće podesiti program GRUB. U datoteku `/boot/grub/menu.lst` potrebno je dodati sljedeći odsječak:

```
title      Debian GNU/Linux - Initng
root       (hd0,0)
kernel     /boot/vmlinuz-2.6.18-5-k7 root/dev/hda1 ro \
init=/sbin/bootchartd_initng
initrd     /boot/initrd.img-2.6.18-5-k7
savedefault
boot
```

Razlika u odnosu na odsječak programa Upstart je, osim naslova, samo u stazi do odgovarajuće Bootchart skripte. Da bi se sustav pokrenuo sa programom Initng, potrebno je odabrati stavku "Debian GNU/Linux – Initng" u izborniku programa GRUB.

3.3. eINIT

Program eINIT [3] je još jedno ostvarenje programa *init*. Inicijalna namjera je bila uključiti i ovaj program u ispitivanje no, nažalost, postupak pokretanja sustava se pokazao problematičnim. Naime, razvojni tim se za sada fokusira na distribuciju Gentoo Linux pa je već kod instalacije programa došlo do komplikacija. Program za izgradnju zahtijeva program Scons za svoju izgradnju. Unatoč tome, program se može instalirati na distribuciji Debian – potrebno je ručno stvoriti simboličke poveznice prema nekim novo instaliranim knjižnicama te u konfiguraciji programa staviti umjesto programa agetty program getty. Program getty služi za tekstualnu (engl. *text mode*) prijavu korisnika na sustav. Prema dokumentaciji programa, nakon ovog koraka se sustav morao moći pokrenuti u tekstualni način. Međutim, to nije bio slučaj – problem je bio negdje u sustavu komunikacije koju koristi program eINIT (riječ je o protokolu P9 prilagođenom za rad na Unixu sličnim sustavima).

Sljedeći pokušaj je bio instaliranje distribucije Gentoo Linux na računalo. Za tu distribuciju postoji već gotovi paket. Međutim, domena početne stranice projekta se promijenila (iz http://www.einit.org u http://einit.jyujin.de), a paket nije bio ažuriran, pa je instalacija na taj način bila nemoguća pošto se sustav za instalaciju paketa kojeg koristi Gentoo temelji na skidanju izvornog koda programa te njegove izgradnje.

4. Ispitivanje odabranih ostvarenja

U ovom poglavlju će se detaljno opisati postupak ispitivanja dosad već odabranih ostvarenja programa *init*. Ispitivati će se ostvarenja izgrađena i podešena u prethodnom poglavlju. Kao pomoć kod ispitivanja, koristiti će se program po imenu Bootchart [17]. Namjena tog programa je vizualno predstavljanje stanja procesâ tijekom pokretanja operacijskog sustava. Postupak instaliranja i podešavanja programa Bootchart je detaljno opisan u dodatku D.

4.1. Pregled ispitne okoline

Ispitivanje će se izvoditi na računalu čija je konfiguracija dana u tablici 4.1. Dani su samo parametri koji mogu bitnije utjecati na brzinu pokretanja operacijskog sustava. Na tome računalu se nalazi distribucija Debian GNU/Linux 4.0r1 (inačica za 32-bitne sustave) koji je instaliran sa pretpostavljenim opcijama.

Tablica 4.1: Konfiguracija ispitnog računala

Parametar	Vrijednost
Procesor	AMD Turion 64 ML-32 (1,8 GHz)
Radna memorija	2×256 MB (dual-channel), DDR 333 Mhz (3,0 ns)
Tvrdi disk	Seagate ST98823A, 5400 min^{-1}, UDMA5

Period vremena koji ovisi od programa *init* počinje pokretanjem programa *init*, a završava pokretanjem posljednje usluge. Ono što program Bootchart ne može je mjerenje vremena potrebnog za gašenja računala – parametra na koji također program *init* utječe. Tako će se to vrijeme morati mjeriti ručno zapornim satom.

Vremena od trenutka paljenja računala do pojave izbornika operacijskih sustava te od trenutka odabira određenog sustava do pokretanja programa *init* ne ovise o programu *init*. Ta se vremena neće mjeriti u testovima. Ipak, mjeriti će se vrijeme od odabira operacijskog sustava do trenutka kad se pokrene program za prijavu na sustav. To je stoga da se može vidjeti koliki udio program *init* ima u vremenu potrebnom da se cjeli sustav pokrene.

4.2. Odabir usluga

Kod odabira usluga koje će se ispitivati, uz pretpostavljene programe koji se pokreću kod pokretanja sustava su instalirani i neki dodatni programi. To su Apache i Postfix poslužitelji, MySQL baza podataka te Slapd (LDAP poslužitelj). Također, sva ostvarenja koja će se ispitivati moraju biti podešena na način da pokreću iste usluge, inače rezultati ispitivanja ne bi bili valjani. Na distribuciji Debian GNU/Linux se po pretpostavci koristi *SysV init*, te je

pretpostavljena razina pokretanja razina broj 2. Kod *SysV init*-a se najprije pokreću razina *S* (sve skripte iz kazala `/etc/rc.S/`), a zatim sve skripte razine 2 (kazalo `/etc/rc.2/`). Upstart koristi iste skripte kod pokretanja, tako da je dovoljno podesiti usluge (skripte) programa *SysV init*. Kod programa Initng, pokreću se sve usluge pretpostavljene razine pokretanja (datoteka `/etc/initng/runlevel/default.runlevel`), kao i sve usluge iz virtualnih razina pokretanja (ako su takve navedene u pretpostavljenoj razini).
Usluge koje će pokretati programi *SysV init* te Upstart u razini *S* vide se na sljedećem ispisu:

```
$ ls /etc/rcS.d/
S02hostname.sh        S04mountdevsubfs.sh   S10chechroot.sh
S11hwclock.sh         S12mtab.sh            S35mountall.sh
S40networking         S45mountnfs.sh        S55bootmisc.sh
S02mountkernfs.sh     S05keymap.sh          S20module-init-tools
S40pcmcia             S48console-screen.sh  S55urandom
S03udev               S07hdparm             S30checkfs.sh
S39ifupdown           S43portmap            S50alsa-utils
```

S druge strane, usluge koje će se pokretati u razini pokretanja 2 su:

```
$ ls /etc/rc2.d/
S10sysklogd     S17mysql-ndb-mgm     S19hplip
S19slapd        S20dbus              S20postfix
S21gdm          S89anacron           S89cron
S99rc.local     S11klogd             S18mysql-ndb
S19mysql        S20cupsys            S20openbsd-inetd
S20ssh          S25bluetooth         S89atd
S91apache2      S99rmnologin
```

Usluge koje će pokretati program Initng su podijeljene u dvije datoteke: `default.runlevel` te `system.virtual`. Sadržaj prve datoteke je:

```
$ cat /etc/initng/runlevel/default.runlevel
runlevel/system
daemon/syslogd
daemon/klogd
daemon/mysql
daemon/hpiod
daemon/slapd
daemon/cupsd
daemon/portmap
daemon/dbus
daemon/hald
daemon/xinetd
daemon/sshd
daemon/gdm
daemon/bluetooth
```

```
daemon/anacron
daemon/atd
daemon/vixie-cron
daemon/apache2
daemon/postfix
system/rmnologin
```

Pretpostavljena razina podiže i sve usluge u virtualnoj razine `system.virtual`. Sadržaj odgovarajuće datoteke je:

```
$ cat /etc/initng/runlevel/system.virtual
system/hostname
system/udev
system/keymaps
system/hdparm
system/clock
system/modules
system/checkfs
system/mountfs
system/ifupdown-debian
system/pcmcia
net/all
system/netmount
system/alsasound
system/bootmisc
system/urandom
daemon/getty
```

4.3. Ispitivanje SysV Init-a

Odabirom stavke "Debian GNU/Linux – SysV init" počinje proces pokretanja sustava sa programom *SysV init*. Na kraju procesa, u kazalu `/var/log/` se nalazi datoteka sa podacima koje je Bootchart skripta prikupila. Potrebno je Java programom bootchart.jar preraditi te podatke u prikladni graf. To se radi na sljedeći način:

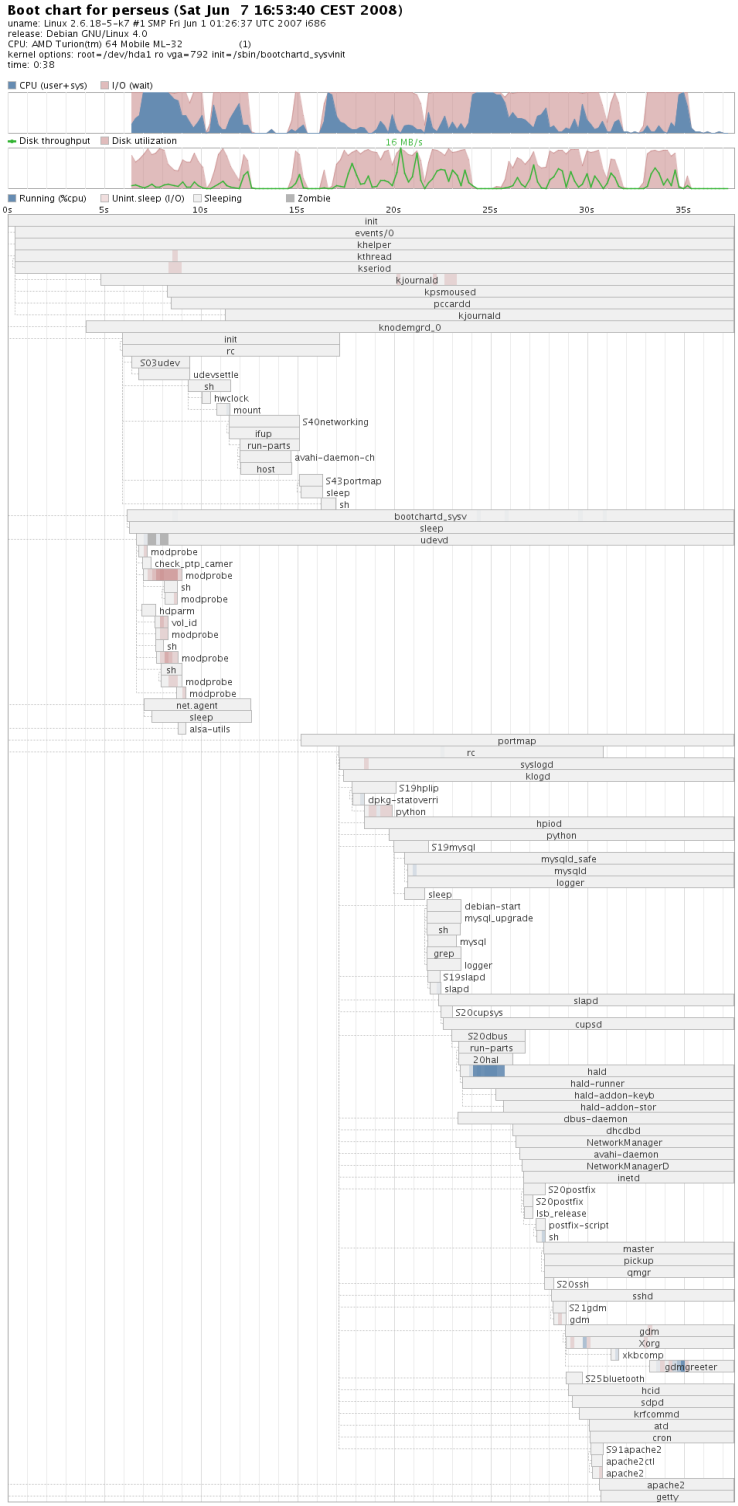
```
$ java –jar /bin/bootchart.jar –f png /var/log/bootchart.tgz
```

Nakon što program završi sa izvođenjem, u tekućem kazalu će se nalaziti datoteka bootchart.png. Datoteka (graf) sa ispitnog sustava je prikazana na slici 4.1.
Karakteristična vremena za program *SysV init* nalaze se u tablici 4.2.

Tablica 4.2: Karakteristična vremena kod ispitivanja programa SysV init

Parametar	Vrijednost [s]
Vrijeme od odabira u programu GRUB do podizanja programa za prijavu	39

Trajanje programa init (izmjereno Bootchartom)	38
Vrijeme zaustavljanja sustava	18



Slika 4.1: Primjer Bootchart grafa

4.4. Ispitivanje Upstarta

Postupak ispitivanja programa Upstart je ekvivalentan ispitivanju *SysV inita*: odabirom stavke "Debian GNU/Linux – Upstart" se sustav počinje pokretati te se na kraju u kazalu `/var/log/` nalazi datoteka sa odgovarajućim podacima. Podaci se na potpuno isti način prerade u graf:

```
$ java –jar /bin/bootchart.jar –f png /var/log/bootchart.tgz
```

Karakteristična vremena za program Upstart nalaze se u tablici 4.3.

Tablica 4.3: Karakteristična vremena kod ispitivanja programa Upstart

Parametar	Vrijednost [s]
Vrijeme od odabira u programu GRUB do podizanja programa za prijavu	36
Trajanje programa init (izmjereno Bootchartom)	23
Vrijeme zaustavljanja sustava	16

4.5. Ispitivanje Initng-a

Neke usluge koje će se koristiti u ispitivanju je potrebno prilagoditi distribuciji Debian GNU/Linux. U usluzi "daemon/slapd" je potrebno promijeniti: stazu do izvršnog programa, stazu do datoteke sa identifikatorom procesa te identifikator korisnika i grupe. Sadržaj izmijenjene usluge je sljedeći:

```
$ cat /etc/initng/daemon/slapd.i
#!/sbin/itype

daemon daemon/slapd {
  need = system/bootmisc;
  suid = root;
  sgid = root;
  exec daemon = /usr/sbin/slapd;
  pid_file = /var/run/slapd/slapd.pid;
  forks;
}
```

Uslugama "daemon/portmap.i" te "daemon/postfix" je na kraj retka koji počinje ključnom riječi "need", a prije znaka ";", potrebno dodati niz "net/eth1".
Potpuno istim postupkom kao i kod prethodna dva programa, dolazi se do karakterističnih vremena za program Initng. Vremena se nalaze u tablici 4.4.

Tablica 4.4: Karakteristična vremena kod ispitivanja programa Initng

Parametar	Vrijednost [s]
Vrijeme od odabira u programu GRUB do podizanja programa za prijavu	28
Trajanje programa init (izmjereno Bootchartom)	18
Vrijeme zaustavljanja sustava	9

4.6. Usporedba rezultata

U tablici 4.5 se nalaze cjelokupni rezultati ispitivanja. Sva vremena izražena su sekundama.

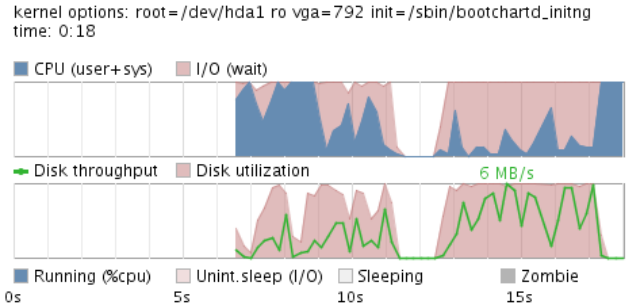
Tablica 4.5: Karakteristična vremena kod ispitivanja programa SysV init

Parametar	SysV init	Upstart	Initng
Vrijeme od odabira u programu GRUB do podizanja programa za prijavu	39	36	28
Trajanje programa init (izmjereno Bootchartom)	38	23	18
Vrijeme zaustavljanja sustava	18	16	9

Kao što se može vidjeti, klasični *SysV init* je (očekivano) najsporiji. S druge strane, Initng se pokazao najbržim u pokretanju sustava. Program Upstart na prvi pogled izgleda jednako sporo kao i *SysV init* no, ako se pogleda bolje, može se vidjeti kako svoj dio posla Upstart obavi zamjetno brže od *SysV inita*. Slična razlika u tim vremenima može se vidjeti i kod programa Initng. To se može objasniti na sljedeći način:

1. ostvarenja programa *init* sa paralelnim pokretanjem procesa vode računa o međuovisnostima,
2. neka usluga se pokreće čim su sve njezine međuovisnosti zadovoljene,
3. broj pokrenutih usluga raste kako se pokretanje sustava privodi kraju – time je sve više međuovisnosti zadovoljeno,
4. na kraju je u stanju pokretanja puno usluga koje koristi iste resurse (procesor, tvrdi disk), a brzina tih resursa onemogućava da se sve usluge izvode brzinom kojom bi se u idealnom slučaju mogle izvoditi.

Takvo razmatranje se može nastaviti na slici 4.2. Slika je isječak iz Bootchart grafa iz ispitivanja provedenog na programu Initng. Dio koji se nalazi na slici je prikaz upotrebe procesorskog vremena te upotrebe tvrdog diska. Jasno je uočljiva velika upotreba te dvije komponente računala potkraj podizanja usluga.



Slika 4.2: Prikaz upotrebe procesora i diska tijekom podizanja sustava programom Initng

Samo paralelnim pokretanjem *SysV init* usluga se dobiva na ubrzanju sustava, no ukoliko se usluge pišu na višoj razini (što program Initng nudi) dobiva se još zamjetnije ubrzanje. To je tako iz dva dva razloga: (1) sam program Initng nudi neke često korištene konstrukte što pojednostavljuje pisanje usluga te su usluge posljedično manje i jednostavnije i (2) jednostavnije usluge mogu izbjeći u potpunosti izvršavanje od strane ljuske sustava što daje dodatno ubrzanje.

Također, program Initng znatno ubrzava gašenje računala.

5. Zaključak

Praktičnom provjerom više ostvarenja je pokazano da se paralelnim pokretanjem usluga proces podizanja operacijskog sustava može značajno skratiti. Iako komponente računala postaju sve brža i kapacitetnija, pojava novih programskih tehnologija uvjetuje sve veći broj usluga koje se moraju pokretati pri pokretanju računala. Dakako, neke tehnologije zastarijevaju, no često se dogodi situacija u kojoj i zastarjele tehnologije moraju biti podržane neko vrijeme radi unazadne kompatibilnosti.

Prednost bržeg pokretanja računala nije samo u kraćem vremenu čekanja već i u manjoj potrošnji energije. To je posebice važno na računalima pogonjenim baterijskim napajanjem gdje to povećava autonomiju rada. U tu kategoriju prvenstveno spadaju prijenosna računala, no ne smije se zaboraviti na razna ugradbena računala u koja spadaju, između ostaloga, i mobiteli. Ugradbena računala, kao i sva druga računala, postaju sve brža te više ne zahtijevaju operacijske sustave posebne namjene kao nekad, već su sposobna izvoditi i operacijske sustave opće namjene prilagođene manjim pokaznicima.

6. Literatura

[1] Initng. Initng Developers. URL: http://www.initng.org/ (26/3/08)

[2] Initng. Wikipedia. URL: http://en.wikipedia.org/wiki/Initng (26/3/08)

[3] eINIT. eINIT Developers. URL: http://einit.org/ (26/3/08)

[4] eINIT. Wikipedia. URL: http://en.wikipedia.org/wiki/EINIT (26/3/08)

[5] Upstart. Canonical. http://upstart.ubuntu.com/ (26/3/08)

[6] Upstart. Wikipedia. http://en.wikipedia.org/wiki/Upstart (26/3/08)

[7] Init. Wikipedia. http://en.wikipedia.org/wiki/Init (26/3/08)

[8] Runlevel. Wikipedia. http://en.wikipedia.org/wiki/Runlevel (26/3/08)

[9] SMF. Wikipedia. http://en.wikipedia.org/wiki/Service_Management_Facility (26/3/08)

[10] Solaris Service Management Facility - Quickstart Guide. BigAdmin System Administration Portal. http://www.sun.com/bigadmin/content/selfheal/smf-quickstart.jsp (26/3/08)

[11] launchd. Apple Inc. http://developer.apple.com/macosx/launchd.html (26/3/08)

[12] launchd. Wikipedia. http://en.wikipedia.org/wiki/Launchd (26/3/08)

[13] Runit. Smarden.org. http://smarden.org/runit/ (10/5/08)

[14] SysV init. Miquel van Smoorenburg. ftp://ftp.cistron.nl/pub/people/miquels/sysvinit/ (13/5/08)

[15] Getting Started. Upstart. http://upstart.ubuntu.com/getting-started.html (23/5/08)

[16] Initng forum. http://forum.initng.org/ (4/6/08)

[17] Bootchart. Bootchart. http://www.bootchart.org/ (6/6/08)

[18] GNU GRUB Manual 0.97. Free Software Foundation. http://www.gnu.org/software/grub/manual/html_node/index.html (6/6/08)

7. Naslov, sažetak i ključne riječi

Naslov:

Paralelizam u pokretanju operacijskih sustava

Sažetak:

Rad se bavi proučavanjem novijih postupaka pokretanja operacijskih sustava u kojima se primjenjuju načela paralelnog pokretanja usluga čime se samo pokretanje nastoji ubrzati. Težište rada je na ispitivanju učinkovitosti postojećih slobodnih i otvorenih rješenja s ciljem da se dokaže, odnosno opovrgne, isplativost takvih postupaka.

Ključne riječi:

ispitivanje, proces, otvoreni sustavi, slobodna programska podrška, operacijski sustavi, init, Unix, GNU, Linux, FreeBSD, MacOS X, Solaris

8. Title, abstract and keywords

Title:

Parallelism in operating system boot process

Abstract:

This thesis addresses issue of recent procedures of booting operating systems where principles of starting services in parallel are used. Faster boot time is key benefit in deployment of these principles. Focus of this thesis is on testing efficiency of existing free and open source implementations, with the purpose of proving, or disproving, their use.

Keywords:

testing, process, open systems, free software, operating systems, init, Unix, GNU, Linux, FreeBSD, MacOS X, Solaris

Dodatak A: Format Upstart poslova

Usluge se kod programa Upstart, u načinu rada bez zadržavanja kompatibilnosti sa programom *SysV init*, nazivaju poslovima (engl. *job*), a nalaze se u kazalu `/etc/event.d/`. To su obične tekstualne datoteke koje sadrže niz ključnih riječi i dodatnih podataka. Naziv pojedinog posla jednak je nazivu pripadne datoteke. Popis ključnih riječi sa objašnjenjima se nalazi u tablici A.1.

Tablica A.1: Ključne riječi poslova programa Upstart

Ključna riječ	Objašnjenje
#	Oznaka komentara. Sve od ove ključne riječi pa do kraja retka se zanemaruje.
`exec` i `script`: jedna od ove dvije ključne riječi mora biti prisutna u opisu pojedinog posla. Tim ključnim riječima se određuje što će se pokrenuti tim poslom.	
exec	Iza ove ključne riječi slijedi puni put do programa u datotečnom sustavu zajedno sa eventualnim argumentima. Ako se koriste posebni znakovi (poput navodnika ili znaka "$") to će rezultirati prosljeđivanjem naredbe ljusci, umjesto da ga Upstart izravno izvrši. Primjer: `exec /bin/foo --opt -xyz foo bar`

script	Označava početak bloka naredbi koje će se redom izvršiti u ljusci. Poziva se ljuska `/bin/sh` (koja može biti simbolička poveznica na npr. `/bin/bash`) sa opcijom `-e`, što znači da će blok naredbi biti prekinut ukoliko bilo koja od naredbi ne uspije. Blok završava sa retkom u kojem se nalazi samo ključna riječ `end script`.
	Primjer:
	`script`
	`  # nešto korisno`
	`  if [ neki_uvjet ]; then`
	`  # nešto korisno`
	`  fi`
	`end script`

`pre-start script` i `post-stop script` nisu obavezne ključne riječi, međutim njima se može obaviti neki posao prije i/ili nakon što je glavni dio posla završen. Tim ključnim riječima se ne može pokrenuti proces, njihova svrha je priprema i čišćenje radne okoline procesa.	

pre-start script	Iza ove ključne riječi slijedi blok naredbi koje će se izvršiti u ljusci, slično kao kod ključne riječi `script`. Blok završava ključnom riječi `end script`. Naredbe koji se nalaze u tom bloku će se izvršiti *prije* glavnog dijela (naredba iza `exec` ili blok naredbi iza `script`). Namjena je pripremanje radne okoline glavnog procesa.
	Primjer:
	`pre-start script`
	`  # pripremi okolinu`
	`  mkdir -p /var/run/foo/`
	`end script`

post-stop script	Sve što je rečeno za ključnu riječ `pre-start script` vrijedi i za ovu ključnu riječ, osim što se blok naredbi izvršava nakon glavnog dijela. Namjena je čišćenje radne okoline glavnog procesa.
	Primjer:
	`post-stop script`
	`  # počisti okolinu`
	`  rm -rf /var/run/foo`
	`end script`

end script	Ovom ključnom riječi se završava blok naredbi započet nekom od sljedećih ključnih riječi: `script`, `pre-start script`, `post-stop script`.
Sa ključnim riječima navedenim do sad je moguće napraviti posao koji će se moći ručno pokrenuti. Gotovo uvijek postoji potreba za automatskim pokretanjem i zaustavljanjem poslova prilikom ulaska u neku, odnosno izlaska iz neke razine pokretanja. Za to Upstart pruža ključne riječi `start on` te `stop on`. Pomoću te dvije ključne riječi posao se registrira za pojedine događaje koje odašilje Upstart. Primjerice, `startup` je događaj koji se odašilje prilikom prvog pokretanja računala dok su `runlevel X`, gdje je X iz skupa 0-6 i *S*, događaji koji se odašilju prilikom promjene pojedine razine pokretanja. Također i sami poslovi odašilju događaje svojim pokretanjem i zaustavljanjem pa se može neki posao podesiti da se pokreće kad se neki drugi posao pokrene ili zaustavi. To se podešava ključnim riječima `started` te `stopped`.	

start on	Iza ove riječi slijedi naziv *jednog* događaja koji će, kad nastupi, uzrokovati pokretanje ovog posla. Ukoliko se posao treba registrirati za više događaja, tada je potrebno navesti više ovih ključnih riječi.
	Primjer:
	`# pokreni kod ulaska u razinu 2`
	`start on runlevel 2`
	`# pokreni kad se pokrene posao tty1`
	`start on started tty1`
	`# pokreni kad se zaustavi posao rcS`
	`start on stopped rcS`

stop on	Slično kao i za `start on`, samo se posao na taj događaj zaustavlja.
	Primjer:
	`# zaustavi kod ulaska u razinu 2`
	`stop on runlevel 2`

console	Ovom ključnom riječi se može promijeniti odredište na koje se ispisuju poruke koje proizvode programi koji se pokreću te izvorište iz kojeg ti programi primaju ulaz. Iza ključne riječi slijedi oznaka kojom se označava izvorište/odredište. Dopuštene oznake su: "output" (kao ulaz i izlaz se koristi `/dev/console`), "owner" (isto kao "output", samo se i signal Control-C šalje procesu) te "none" (ovo je pretpostavljeno, kao ulaz i izlaz se koristi `/dev/null`, tj. sav ispis se zanemaruje).
	Primjer:
	`exec echo primjer`
	`console output`

Ručno pokretanje usluga kod programa *SysV init* je vrlo jednostavno pošto je riječ o najobičnijim skriptama ljuske. Ako se Upstart koristi bez *SysV init* kompatibilnosti te se koriste njegovi poslovi, tada te poslove nije moguće izravno pokretati. Pošto ručno pokretanje usluga nije rijetkost, razvijeni su alati koji to omogućavaju. Postoje tri jednostavnija programa (`start`, `stop` i `status`) te jedan složeniji (`initctl`) kojim se može napraviti sve što se može napraviti sa jednostavnija tri programa pa i više.

Alati `start` i `stop` primaju kao parametar ime posla kojeg se želi pokrenuti, odnosno zaustaviti. Na ekran se ispisuje promjena stanja koja pritom nastupa. Alat `status` prima kao argument ime posla te ispisuje njegovo trenutno stanje. Primjer korištenja spomenutih alata je dan u nastavku.

```
# status tty1
tty1 (stop) waiting
# start tty1
tty1 (start) running, process 4622
# status tty1
tty1 (start) running, process 4622
# stop tty1
tty1 (stop) running, process 4622 killed
```

Ispis tih alata se čita na sljedeći način: posao (`tty1`) je do sad bio pokrenut/zaustavljen ("start"/"stop"), sada je u stanju "running"/"waiting", a identifikator mu je 4622.

Alat `initctl` osim ovih mogućnosti dodatno omogućava i sljedeće:

- ispis trenutno poznatih poslova te njihova trenutna stanja (naredba "list"),

- odašiljanje događaja i time utjecanje na stanja poslova (naredba "trigger"),
- ispis svih promjena stanja poslova na terminal (naredba "jobs"),
- ispis svih odaslanih događaja na terminal (naredba "events"),
- gašenje računala odašiljanjem događaja "shutdown" te neposredno nakon toga odašiljanje nekog drugog događaja te
- slanje proizvoljnog, korisnički definiranog, događaja procesima (naredba "emit").

Potonje će biti pokazano na primjeru. Pretpostavimo da imamo sljedeći posao:

```
$ cat /etc/event.d/pingpong
on ping
exec echo pong
console output
```

Kao što se može vidjeti, sve što ovaj posao radi jest ispisivanje niza "pong" na terminal kada se pojavi događaj "ping" (koji je izmišljen, nije ugrađen u Upstart). Sljedeća naredba će pokrenuti taj posao:

```
# initctl emit ping
# pong
```

Većina dodatnih naredbi alata `initctl` služi za otkrivanje pogrešaka u programu Upstart, no mogu biti korisne i administratorima.

U nastavku će biti u cijelosti objašnjen jedan primjer posla. Riječ je o poslu naziva "rc6" koji je zadužen za ponovno pokretanje računala. Taj posao se nalazi u arhivi example-jobs-0.3.9.tar.gz koja je bila instalirana nakon programa Upstart. Sadržaj posla je dan u nastavku.

```
$ cat /etc/event.d/rc6
# rc6 - runlevel 6 compatibility
#
# This task runs the old sysv-rc runlevel 6 ("reboot") scripts.

start on reboot
start on runlevel-6

stop on shutdown
stop on runlevel-2
stop on runlevel-3
stop on runlevel-4
stop on runlevel-5

script
  set $(runlevel || true)
```

```
    if [ "$2" != "0" ] && [ "$2" != "6" ]; then
       set $(runlevel --set 6 || true)
    fi

    if [ "$1" != "unknown" ]; then
       PREVLEVEL=$1
       RUNLEVEL=$2
       export PREVLEVEL RUNLEVEL
    fi

    exec /etc/init.d/rc 6
  end script
```

Kada se pojavi jedan od događaja "reboot" ili "runlevel-6", počinje se izvršavati blok između ključnih riječi `script` te `end script`. Nasuprot tome, ako se pojavi neki događaj koji označava ulazak u neku drugu razinu pokretanja (događaj "shutdown" je istoznačnica za razinu 0), a ovaj posao je bio pokrenut, tada se njegovo izvođenje prekida.

U samoj skripti, naredba `set` postavlja varijable ljuske `$1` te `$2`: u varijabli `$1` se nalazi kôd prethodne razine pokretanja dok se u varijabli `$2` nalazi kôd trenutne razine. Ukoliko se u varijabli `$1` nalazi niz "unknown", tada varijabla `$2` ne sadrži ništa, tj. informacija o trenutnoj razini pokretanja ne postoji. To se dogodi npr. ako izvršimo tu naredbu za vrijeme dok je pokrenuto neko drugo ostvarenje programa *init*, a ne Upstart. Nakon što su spomenute varijable postavljene, provjerava se sadržaj varijable `$2` te se, ukoliko trenutna razina nije 0 niti 6 (tj. gašenje ili ponovno pokretanje računala), trenutna razina pokretanja se postavlja u razinu 6, tj. razinu ponovnog pokretanja računala. Zatim, ukoliko je prethodna razina poznata, postavljaju se varijable `$PREVLEVEL` te `$RUNLEVEL` te se izvezu tako da budu varijable okoline (engl. *environment variable*, postiže se naredbom `export`). To je potrebno iz razloga što klasične *SysV init* skripte očekuju postavljenost tih varijabli, a ovaj posao naredbom `exec` izvršava skriptu `rc` sa parametrom 6, što znači da će se izvršiti sve *SysV init* skripte za razinu 6. Zaključuje se da je posao pisan s ciljem da zadrži tu, već nekoliko puta spomenutu, kompatibilnost sa programom *SysV init*.

Dodatak B: Format Initng uslugâ

Initng usluge se nalaze u kazalu `/etc/initng/`. Dalje su one podijeljene u podkazala `daemon/`, `net/`, `service/` te `system/`. U kazalu `daemon/` se nalaze usluge programa koji se pokreću kao pozadinski procesi te obavljaju usluge bez interakcije sa korisnikom. U kazalu `net/` se nalaze usluge vezane uz podešavanje mrežnih sučelja dok se u kazalima `service/` te `system/` se nalaze preostale usluge.

Initng podržava dva formata usluga – tzv. *i*-format te XML format. Pošto je *i*-format još uvijek glavni format usluga programa Initng, ovdje će se pažnja posvetiti upravo njemu. Initng-ove *i*-usluge su, kao i kod Upstartovih poslova, obične tekstualne datoteke koje sadrže niz ključnih riječi i dodatnih podataka. Popis ključnih riječi sa objašnjenjima se nalazi u tablici B.1.

Tablica B.1: Popis ključnih riječi i-usluga programa Initng

Ključna riječ	Objašnjenje
#	Oznaka komentara. Sve iza ove ključne riječi se zanemaruje.
service	Ključna riječ iza koje slijedi naziv usluge te blok unutar kojeg se nalaze ostale ključne riječi usluge. U jednoj datoteci može biti više ovih ključnih riječi – time i više usluga. Blok počinje znakom "{", a završava znakom "}". Kod izgradnje međuovisnosti usluga, reference se odnose na ove nazive, a ne nazive datoteka.
Ključne riječi usluga	
exec	Koristi se zajedno sa ključnom riječi "start", služi za izvršavanje jedne naredbe sustavskim pozivom `exec()`.
script	Koristi se zajedno sa ključnom riječi "start", služi za izvršavanje niza naredbi u ljusci.

start	Ključna riječ iza koje slijedi blok naredbi koje će se izvršiti prilikom pokretanja usluge. Blok naredbi će se izvršiti kao skripta ljuske.
	Primjer:
	`# pokretanje jedne naredbe`
	`exec start = /bin/foo --opt -xyz foo bar`
	`# pokretanje niza naredbi`
	`script start {`
	`  # nešto korisno`
	`  if [ neki_uvjet ]; then`
	`  # nešto korisno`
	`  fi`
	`};`

stop	Identično ključnoj riječi "start", s iznimkom što se naredba (ili blok) koji slijedi izvršava kod gašenja usluge.
Ključne riječi pozadinskih procesa	

daemon	Koristi se na isti način kao ključne riječi "start" i "stop", ali se koristi kada proces ne završava (pozadinski proces). Kada se koristi ova ključna riječ, Initng ne čeka na završetak procesa – ukoliko proces ipak završi, Initng će smatrati da se program nije uspio pokrenuti te će ga eventualno ponovno pokušati pokrenuti.
respawn	Ako se iza ove ključne riječi nalazi riječ "yes", tada će se program/blok iza ključne riječi "daemon" pokrenuti ponovno ukoliko prethodno pokretanje nije uspjelo. Druga mogućnost je da se navede riječ "no".
pid_file	Neki pozadinski procesi i nakon uspješnog pokretanja vraćaju izlaznu vrijednost. Kako bi se spriječilo Initng da ponovno pokreće tu uslugu, zadaje se ova ključna riječ iza koje slijedi puni put do datoteke u kojoj se nalazi identifikator tog pozadinskog procesa. U jednoj usluzi može biti i više ovih ključnih riječi, u tom će slučaju Initng pogledati sve datoteke prije nego počne smatrati program neuspjelim.. Primjer: `pid_file = /var/run/smartd.pid;`
forks	Noviji način rješavanja problema sa pozadinskim procesima. Moguće je više vrijednosti: ◆ "yes" – pozadinski proces stvara novi proces (engl. *fork*) te vraća neku vrijednost, ◆ "no" – pozadinski proces ne stvara novi proces, ◆ "noret" – pozadinski proces stvara novi proces, ali roditeljski proces nastavlja s radom sve dok proces dijete ne završi. Ukoliko se ova ključna riječ ne navede, a koristi se ključna riječ "pid_file", podrazumijevati će se vrijednost "yes", inače vrijednost "no".
Ključne riječi zajedničke pozadinskim procesima i uslugama	
need	Usluga zahtijeva neku drugu uslugu. Usluga koja se zahtijeva mora biti pokrenuta prije promatrane usluge. Primjer: `need = system/initial;`
use	Usluga koristi neku drugu uslugu. Usluga koja se zahtijeva mora ili biti pokrenuta ili se mora trenutno pokretati u trenutku pokretanja promatrane usluge. Primjer: `use = daemon/mysql;`
conflict	Usluga ne smije biti pokrenuta istovremeno sa nekom od navedenih usluga.

provide	Pošto često postoji više ostvarenja programa neke namjene, da se ne bi u sve usluge koje ovise o nekom od tih programa morale stvarati međuovisnosti prema svakom od njih, stvaraju se nestvarne (virtualne) usluge. Ostale usluge tada trebaju ovisiti samo o toj virtualnoj usluzi. Primjer: `provide virtual/dm;`
require_network	Prije pokretanja promatrane usluge mora biti podignuto barem jedno mrežno sučelje osim lokalne petlje (engl. *loopback interface*).
exec_path	Postavlja moguće staze do programa koji se izvršava. Ako se u jednoj ne nalazi program, gleda se sljedeća i td. Koristi se uz ključne riječi "start", "stop" i "daemon". Primjer: `exec_path daemon = /usr/bin/gdm /usr/sbin/gdm;`
exec_args	Predaju se argumenti programa zadanog pored ključne riječi "exec_path". Primjer: `exec_args daemon = –nodaemon;`
env	Postavlja varijablu okoline. Može biti više ovih ključnih riječi u jednoj usluzi. Primjer: `env FOO = bar;`
env_file_required	Učitava datoteku u kojoj su popisane varijable okoline te ih postavlja. Ako datoteka ne postoji, usluga se neće podignuti. Ako ova riječ postoji, sve pojave ključne riječi `env` se zanemaruju.
env_file	Slično kao `env_file_required`, osim što usluga uvijek uspijeva. Ako ova riječ postoji, sve pojave ključnih riječi `env` i `env_file_required` se odbacuju.
nice	Postavlja prioritet procesa. Primjer: `nice = –4;`
suid	Postavlja korisnički identifikator procesa. Primjer: `suid = root;`
sgid	Postavlja identifikator grupe na proces. Primjer: `sgid = wheel;`
delay	Postavlja vremensku odgodu (u sekundama). Primjer: `delay = 5;`

chdir	Mijenja radno kazalo prije nego se proces pokrene. Primjer: `chdir = /home/korisnik/;`
chroot	Maskira korijenski put (`/`) za proces koji će se pokrenuti. Koristi se za sigurnosne potrebe. Primjer: `chroot = /usr/local/;`

stdout	Preusmjerava standardni izlaz iz procesa u datoteku ili uređaj. Najčešće se radi o preusmjeravanju u uređaj `/dev/null`.
	Primjer:
	`stdout = /dev/null;`
	`stdout = /var/log/process.log`

stderr	Slično riječi `stdout`, samo se preusmjerava standardni izlaz za greške (stderr). Primjer: `stderr = /dev/null;`
stdall	Preusmjerava i standardni izlaz i izlaz za greške u datoteku ili uređaj.

Nekoliko napomena uz pisanje usluga:

- Kada se pokreće blok naredbi (ključna riječ "script start"), izlazni kôd generiran tijekom izvođenja tog bloka utječe na uspješnost pokretanja usluge. Na primjer, ako se unutar bloka izvede `exit 1`, Initng će smatrati da se usluga nije uspjela pokrenuti.
- Preporuka je da se koristi ključna riječ "exec" radije nego "script" kad god je to moguće. Razlog tomu je što je izvođenje prilikom korištenja riječi "exec" puno brže nego kod korištenja riječi "script". Kod riječi "exec" se koristi sustavski poziv `exec()` dok se kod riječi "script" blok naredbi koji slijedi izvršava u ljusci sustava.
- Trebalo bi težiti kompatibilnosti sa POSIX specifikacijom. Primjerice, umjesto pisanja preusmjeravanja oblika `&>/dev/null` treba pisati `>/dev/null 2>&1`.

Sa programom Initng dolazi nekoliko alata koji olakšavaju njegovo upravljanje. Pošto usluge programa Initng nisu skripte ljuske, ne možemo neku uslugu pokrenuti jednostavnim izvršavanjem te usluge iz ljuske. Zbog toga je nužno koristiti odgovarajući alat za ručno pokretanje i zaustavljanje usluga. Kod programa Initng je to alat `ngc`. Popis najvažnijih opcija sa opisima je dan u tablici 8.1.

Tablica 8.1. Popis opcija alata ngc

Opcija	Opis
-s ili --status	Ispisuje sve usluge koje su se pokretale tijekom pokretanja sustava te njihovo trenutno stanje.

-u ili --start	Pokreće uslugu koja se navodi kao argument iza ove opcije.
-d ili --stop	Zaustavlja uslugu koja je navedena kao argument iza ove opcije.
-z ili --zap	Kada neka usluga zaglavi, potrebno je pokrenuti ovu opciju (sa nazivom te usluge kao argumentom) te se tek nakon toga može ta usluga ponovno pokrenuti.
-r ili --restart	Ponovno pokreće uslugu koja se navodi kao argument iza ove opcije.
-6 ili --reboot	Ponovno pokreće sustav. Broj 6 je samo asocijacija na broj razine pokretanja u programu *SysV init*, nema nikakve veze sa tim programom.
-0 ili --poweroff	Gasi računalo. Vrijedi napomena uz prethodnu opciju.

Dakle, za pokretanje usluge "daemon/cupsd", potrebno je izvršiti sljedeće:

```
# ngc –u daemon/cupsd
```

U nastavku se nalazi opis jedne usluge Initng programa. Riječ je o usluzi koja pokreće program SLiM. Taj program služi za grafičku prijavu korisnika na sustav (engl. *Display Manager*).

```
$ cat /etc/init.d/daemon/slim.i
#!/sbin/itype
# This is a i file, used by initng parsed by install_service

# NAME: SLIM
# DESCRIPTION: The "slim" desktop-independent login manager.
# WWW: http://slim.berlios.de/

daemon daemon/slim {
   need = system/bootmisc;
   conflict = daemon/kdm daemon/wdm daemon/xdm daemon/gdm;
   provide = virtual/dm;
   use = system/modules system/coldplug;
   exec daemon = /usr/bin/slim –d;
}
```

Sve Initng usluge u *i*-formatu imaju u prvom retku oznaku `#!/sbin/itype`, što znači da bi se ta skripta izvršila pomoću programa `/sbin/itype` kada bi se izvršavala izravno iz ljuske. Međutim, taj program ne postoji te se ta oznaka stavlja samo radi konvencije.
Nakon nekoliko komentara se nalazi blok koji pripada usluzi. Može se vidjeti da usluga ovisi o usluzi "system/bootmisc" – ta usluga osigurava viši stupanj pokrenutosti sustava. Na primjer, prijava na sustav nema smisla ako datotečni sustavi nisu spojeni te ako nije podignuto lokalno mrežno sučelje (zbog naravi grafičkog poslužitelja X).

Pošto je SLiM samo jedan između mnogih programa te namjene, mora se osigurati da dva ili više takva programa nisu pokrenuta istovremeno. To se osigurava ključnom riječi "conflict". Iz istog razloga postoji i ključna riječ "provide". Da se ne bi u sve usluge koje ovise o programu za grafičku prijavu na sustav morale navoditi međuovisnosti prema svakom od ostvarenja tog programa, svaki od tih programa pruža nestvarnu uslugu "virtual/dm" pa je dovoljno stvoriti međuovisnost prema toj usluzi.
Na kraju, izvršava se program SLiM kao pozadinski proces.

Dodatak C: Uvod u program GRUB

GNU GRUB [18] je program za pokretanje operacijskih sustava (engl. *bootloader*). Postoji više programa te namjene (primjerice LILO), no GRUB se koristi na svim većim distribucijama. Osim pokretanja Linuxa, ima mogućnost pokretanja i drugih operacijskih sustava – ako ne izravno onda pokretanjem ekvivalentog programa koji dolazi sa tim drugim operacijskim sustavom. U izvornoj literaturi se to naziva *chainloader*.

Konfiguracijska datoteka programa GRUB je `/boot/grub/menu.lst`. U toj datoteci se nalazi, osim općenitih postavki, popis operacijskih sustava između kojih će korisnik moći birati nakon paljenja računala. Kao što je već spomenuto, sama jezgra sustava pokreće program *init*. Međutim, Linux jezgri se može prenijeti odgovarajući parametar koji utječe na program koji će jezgra pokrenuti. Parametar ima oblik: "init=/put/do/programa". Pošto, sa stajališta jezgre, nema razlike između programa *init* i bilo kojeg drugog programa, preko tog parametra se može pokrenuti na primjer ljuska.

Objašnjenja pojedinih ključnih riječi dana su u tablici C.1.

Tablica C.1: Popis najvažnijih ključnih riječi programa GRUB

Ključna riječ	Objašnjenje
title	Tekst koji slijedi će pisati u izborniku programa GRUB nakon paljenja računala.
root	Iza ove ključne riječi slijedi oznaka particije na kojoj se nalazi jezgra operacijskog sustava. Oznaka particije je svojstvena programu GRUB. U oznaci *(hd0,0)*, *hd0* označava da je riječ o glavnom disku na prvom IDE kanalu (engl. *primary master*), dok druga nula označava da je riječ o prvoj particiji na tom disku.
kernel	Iza ove riječi slijedi puni put do slike jezgre, iza čega slijede parametri koji se prosljeđuju jezgri. Parametar "root" označava put do blok-uređaja (engl. *block device*) koji predstavlja korijensku particiju (engl. *root partition*). Parametar "ro" označava da će se korijenska particija spojiti bez mogućnosti pisanja u nju (engl. *read-only*), dok parametar "init" označava puni put do programa *init*.
initrd	Iza ove riječi slijedi puni put do ranije spomenute *initrd/initramfs* datoteke.
savedefault	Sprema trenutni zapis kao glavni zapis.
boot	Pokreće operacijski sustav.

Dodatak D: Postavljanje programa Bootchart

Program Bootchart je namijenjen vizualnom predstavljanju stanja procesâ tijekom pokretanja operacijskog sustava. Program se sastoji iz dva dijela: prvi dio je skripta ljuske koja je smještena u kazalo `/sbin/`, a drugi dio je Java program (u obliku JAR datoteke). Skripta je namijenjena praćenju stanja procesa tijekom pokretanja operacijskog sustava te zapisivanja prikupljenih podataka u datoteku, dok Java program iz tako stvorene datoteke stvara sliku u nekom od podržanih formata (PNG, SVG ili EPS).

Prije instalacije programa, potrebno je instalirati neko od razvojnih okruženja za Java programski jezik, npr. Free Java SDK te paket Apache Ant:

```
# apt-get install free-java-sdk ant
```

Nakon zadovoljavanja potrebnih preduvjeta, može se krenuti sa izgradnjom programa Bootchart. Postupak izgradnje i postavljanja programa Bootchart je sljedeći:

1. Sa stranice projekta [17] se dohvati arhiva sa izvornim kodom programa:

   ```
   $ wget
   http://prdownloads.sourceforge.net/bootchart/bootchart-0.9
   .tar.bz2
   ```

2. Raspakira se arhiva te se promijeni radno kazalo u novo stvoreno kazalo:

   ```
   $ tar xf bootchart-0.9.tar.bz2
   $ cd bootchart-0.9/
   ```

3. Izvrši se skripta `install.sh` – skripta će instalirati skriptu `bootchartd` u kazalo `/sbin/`, te konfiguraciju programa (`/etc/bootchartd.conf`):

   ```
   # ./install.sh
   ```

4. Pokrene se program `ant` – time će se izgraditi Java paket `bootchart.jar`:

   ```
   $ ant
   ```

5. Opcionalno, novo stvoreni paket se može iskopirati na prikladno mjesto (neko od kazala koje se nalazi u varijabli okoline `PATH`):

   ```
   # cp bootchart.jar /usr/bin/
   ```

Konfiguracijska datoteka koja pripada programu Bootchart je `/etc/bootchartd.conf`. Pretpostavljene vrijednosti u toj datoteci su dovoljne, no može se, na primjer, promijeniti

smještaj datoteke sa stanjima procesa ili pretpostavljeni izlazni format slike. Ispis konfiguracijske datoteke korištene za potrebe ovoga dokumenta se nalazi ispod.

```
$ cat /etc/bootchartd.conf
TMPFS_SIZE=32m
BOOTLOG_LOCK=".lock"
SAMPLE_PERIOD=0.2
PROCESS_ACCOUNTING="no"
BOOTLOG_DEST=/var/log/bootchart.tgz
AUTO_RENDER="no"
AUTO_RENDER_FORMAT="png"
AUTO_RENDER_DIR="/var/log"
```

Program Bootchart se koristi tako da se u odgovarajući odsječak programa GRUB preko parametra "init" prenese staza do skripte `bootchartd`. Skripta `bootchartd` bi, prema dokumentaciji, trebala, kada obavi svoj inicijalizacijski dio posla, pokrenuti program koji se nalazi iza dodatnog parametra "bootchart_init", ili pretpostavljeni init (`/sbin/init`) ukoliko spomenuti parametar nije prenesen. Međutim, ma koji program se prenese tim parametrom, Bootchart uvijek pokreće pretpostavljeni program *init*. Taj se problem može riješiti tako da se napravi kopija skripte `bootchartd` te da se u njoj zamijeni pretpostavljeni program *init* sa željenim. Za program Upstart dovoljno je napraviti sljedeće:

```
# cd /sbin/
# cat bootchartd | sed 's:/sbin/init:/sbin/initng:g' >
bootchartd_upstart
```

Slično se napravi i za program Upstart. U konfiguraciji programa GRUB se jezgri preko parametra "init" prenese staza do odgovarajuće Bootchart skripte.

Nakon što odgovarajuća `bootchartd` skripta pokrene program *init*, periodički se bilježi stanje procesa. To vrijeme je određeno vrijednošću iza ključne riječi "SAMPLE_PERIOD" u konfiguracijskoj datoteci programa.

Kada se sustav pokrene, u odgovarajućoj datoteci (ključna riječ "BOOTLOG_DEST") se nalaze u .tar.gz arhivi parametri koje je program zabilježio. Ti se podaci prerade Java programom `bootchart.jar` u neki od podržanih formata slike. Format se može odrediti opcijom `-f`, podržani formati su PNG, EPS te SVG. Na primjer, za format slike PNG potrebno je na sljedeći način pokrenuti program:

```
$ java -jar /usr/sbin/bootchart.jar -f png bootchart.tgz
```