Deskriptorji: SISTEM PORAZDELJENI, VODENJE, UPRAVLJANJE, SISTEM OPERACIJSKI

Jože Rugelj
IJS Ljubljana

POVZETEK Upravljanje omogoča ohranjanje konsistentnosti sistema in pomaga uporabnikom pri delu s sistemom. Pomen upravljalskih aktivnosti se povečuje z naraščanjem zmogljivosti in kompleksnosti sistemov in z njihovo porazdeljenostjo. Najpomembnejše upravljalske funkcije so upravljanje konfiguracije, spremljanje delovanja sistema, ravnanje ob odpovedih, varnostne in zaščitne aktivnosti, obračunavanje in upravljanje z imeni.

ABSTRACT Management enables keeping the consistency of a system and facilitates the usage of a system. The importance of management activities increases with the growing of performances and complexity of the systems and with their distribution. The most important management functions are configuration management, monitoring, fault handling, protection and security activities, accounting and name management.

Upravljanje porazdeljenih sistemov vključuje mnogo vidikov porazdeljenega procesiranja in je v enakem odnosu do porazdeljenih aplikacij kot klasičen operacijski sistem do lokalnih aplikacij [COST83]. Zato pogosto imenujemo nabor upravljalskih funkcij za porazdeljen sistem porazdeljen operacijski sistem. Najbolj splošno bi lahko kot cilj upravljanja porazdeljenega sistema opredelili ohranjanje sistema v stanju, ki ga zahtevajo uporabniki, in to s čimmanjšimi stroški. To pomeni, da sistem omogoča uporabnikom in upravljalcem sistema načrtovanje, organiziranje, nadzorovanje in vodenje uporabe porazdeljenih virov informacijskega procesiranja.

1 Cilji upravljanja

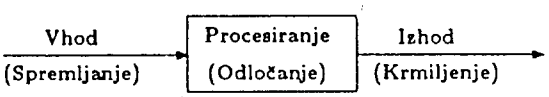
Upravljanje sistema se izvaja v skladu z upravljalsko politiko, ki določa osnovna pravila v zvezi z upravljalskimi odločitvami [Slom87]. Problemi, s katerimi se ukvarja upravljalska politika, so komercialne in tehnične narave. V podjetjih je njen cilj maksimizacija prihodkov, v izobraževalnih ustanovah minimizacija stroškov in pri vojaških projektih zagotovljena visoka zanesljivost. Upravljalsko politiko uporabljamo za določitev strategije in taktike pri upravljanju. Strategija je dolgoročen načrt, kaj je treba storiti za uresničenje ciljev, taktika pa nam pove, kako bomo to storili.

Upravljalske aktivnosti bi lahko razdelili na

- dolgoročne
- operativne.

Dolgoročne odločitve se nanašajo predvsem na načrtovanje razvoja in širjenja porazdeljenega sistema, pa tudi na strategije optimizacije uporabe virov, poimenovanja virov, varnosti in zanesljivosti sistema. Take dolgoročne odločitve sprejemajo *ljudje, ki vodijo in upravljajo porazdeljen sistem*. V primeru tesno sklopljenih sistemov, ki jih obravnavamo v tem delu, pri tem ni posebnih težav, saj je lastnik porazdeljenega sistema ena organizacija. Zato ne prihaja do konfliktov pri reševanju teh problemov tako kot pri šibko sklopljenih sistemih.

Operativno upravljanje porazdeljenega sistema se izvaja v času delovanja sistema, torej sočasno z izvajanjem uporabniških programov. Operativno upravljanje skrbi za optimizacijo izrabe virov v sistemu in možnost spreminjanja konfiguracije sistema, za reševanje iz napak, za varnost sistema in za obračunavanje stroškov za uporabo virov. Večina teh funkcij je soodvisna med seboj. Odločitve morajo biti zelo hitre, v nekaj desetinkah ali tisočinkah sekunde in zato je skoraj nemogoče, da bi pri njih sodeloval človek. Upravljalske aktivnosti so zato izvedene kot strežniki znotraj sistema. Pogosto upravljalske aktivnosti niso eksplicitno ločene in delujejo kot vključene komponente v standardnih strežnikih. Spremljajo delovanje posameznih komponent in sproti ocenjujejo njihovo delovanje. V primeru, ko delovanje komponente ne ustreza pričakovanjem, strežniki upravljalci izvedejo predvidene operacije [COST83]. Slika .1 prikazuje splošno shemo upravljalske operacije.



Slika .1: Shema upravljalske operacije

Samo v skrajnih primerih ali na zahtevo upravljalske funkcije javljajo rezultate svoje aktivnosti človeku, upravljalcu in uporabniku sistema.

Oblika in obseg informacije, ki jo pripravi upravljalska aktivnost, je odvisna od tega, komu je namenjena. Uporabnika predvsem zanimajo osnovne informacije o stanju sistema, ki naj bodo kratke in pregledne. Za upravljalce, ki skrbijo tudi za vzdrževanje sistema in dolgoročno upravljanje, pa so pomembne vse podrobnosti.

2 Zgodovinski pregled upravljalskih pristopov v računalniških sistemih

Glavna cilja upravljanja računalniških sistemov sta čimboljša izraba računalniške opreme in čimbolj enostaven vmesnik proti

3.2 Spremljanje delovanja sistema

Spremljanje delovanja sistema je povezano z delovanjem vseh upravljalskih funkcij, saj služi kot vhodni podatek za proces odločanja in izvajanja upravljalskih aktivnosti. Ta funkcija upravljanja torej nudi servis ostalim upravljalskim funkcijam, redkeje pa neposredno uporabniku. Poseben pomen imajo rezultati spremljanja delovanja sistema za ravnanje ob odpovedih in reševanje iz napak. Funkcija spremljanja delovanja sistema je lahko realizirana centralizirano ali porazdeljeno. Tudi pri tej funkciji imata ena in druga rešitev dobre in slabe lastnosti. Centralizirana rešitev je enostavnejša, a manj zanesljiva in prilagodljiva, porazdeljena pa ima komplementarne lastnosti. Pri obeh se pojavlja problem konsistentnosti stanja sistema, ki je posledica nedeterminizma pri prenosu sporočil po komunikacijski mreži in sinhronizacijskih problemov, ki sledijo iz tega. Pri spremljanju delovanja se zbere velike količine podatkov in strežnik, ki opravlja funkcijo spremljanja, mora te podatke zbrati in urediti tako, da nudijo ustrezno informacijo o delovanju sistema.

3.3 Ravnanje pri odpovedih

Ravnanje pri odpovedih je upravljalska funkcija, ki nudi servis strežnikom in uporabnikom porazdeljenega sistema, ko pride do odpovedi ali nepravilnega delovanja ene ali več komponent v sistemu. Posledica nenormalnega stanja sistema ali kateregakoli njegovega dela se pokaže v obliki napak pri izvajanju operacij pri nekaterih strežnikih v sistemu. Ravnanje pri odpovedih vključuje zaznavo napak in njihovo analizo, iskanje okvarjenih komponent, odpravo okvare ali ustrezno zamenjavo komponente, odpravljane posledic napak in obvestila o napaki upravljalcem sistema in po potrebi uporabnikom. V porazdeljenem sistemu so problemi ravnanja z odpovedmi še večji, saj se napake zaradi sodelovanja med porazdeljenimi komponentami hitro širijo, nadzor in odpravljanje posledic pa sta zaradi porazdeljenosti težja [LeLa81].

Pri pregledu aktivnosti se bomo posvetili tistim servisom, ki jih v primeru odpovedi nudijo strežniki v sistemu.

Odkrivanje napak

Če hočemo odkrivati napake v sistemu med delovanjem, moramo vnaprej poznati vsaj približno obnašanje sistema. Tako lahko v vsakem trenutku primerjamo dejansko stanje sistema s predvidenim in če se pojavi odstopanje, to pomeni napako. Ker pa v splošnem stanj sistema ne poznamo vnaprej, dodamo posebne komponente strojne in programske opreme, ki izvajajo različne teste ob začetku delovanja sistema in med samim delovanjem, periodično ali ob posebnih pogojih. Rezultate teh testov pri pravilno delujočem sistemu poznamo in zato lahko odkrijemo odpovedi in nepravilnosti, če se pojavijo. Taki dodatni testi sicer zmanjšujejo zmogljivost sistema, vendar je to cena, ki jo moramo plačati za povečano zanesljivost delovanja. Pogostost in natančnost testiranja je odvisna od zahtev po zanesljivosti. Tudi zmanjšanje zmogljivosti sistema je lahko znak za napake v sistemu, vendar je take značilnosti razmeroma težko formalno opisati in zato tudi niso primerne za uporabo pri sprotni diagnostiki.

Analiza napak in odkrivanje okvarjenih komponent

Analiza rezultatov testov pomaga pri odkrivanju napak in vzrokov, zaradi katerih je prišlo do napak. V porazdeljenih sistemih je težko določiti izvor napake, saj se rezultati prenašajo med posameznimi strežniki, ki so na različnih lokacijah. Napako lahko zaradi premalo pogostega testiranja spregledamo in jo zaznamo šele, ko se je že razširila po sistemu. Zato kljub očitni napaki težko odkrijemo njen izvor. Sodelovanje strežnikov, ki se ukvarjajo z analizo napak in so porazdeljeni po sistemu, omogoča analizo dogajanj v celotnem sistemu.

Popravilo ali zamenjava okvarjene komponente

Ko odkrijemo vzrok za napako in s tem tudi komponento, ki ne deluje pravilno, najprej tako komponento izločimo, da ne bi povzročala dodatnih težav. To storimo tako, da o napaki obvestimo upravljalca konfiguracije. Potem poskušamo tako komponento delno ali v celoti popraviti s servisi, ki so na razpolago v sistemu. Če tako popravilo uspe, komponento vrnemo v sistem. Pri komponentah, ki so bolj kompleksne, je možno tudi delno popravilo. V takih primerih komponenta izvaja samo omejeno število funkcij. Kadar okvare ne moremo odpraviti s servisi, ki so na razpolago v sistemu, ali jo lahko le delno odpravimo, o tem obvestimo upravljalca sistema; le-ta o okvari obvesti vzdrževalce programske ali strojne opreme in po potrebi tudi uporabnika. To je potrebno predvsem v primerih, ko servis zaradi odpovedi komponent ne more nuditi predvidenega servisa, saj določeni viri niso replicirani. Če se zaradi odpovedi posamezne komponente delo porazdeli med druge in to pomeni samo zmanjšanje zmogljivosti sistema, to običajno ni usodno za uporabnika in ga zato ni treba vznemirjati s poročili o odpovedih.

Odprava posledic odpovedi

Odpravljanje posledic odpovedi po odstranitvi ali popravilu okvarjene komponente je osredotočeno predvsem na razveljavitev rezultatov, ki vsebujejo napake, posledice odpovedi. Kompleksnost problema lahko zmanjšamo z uvedbo principa nedeljivih akcij. To so segmenti programske opreme, ki se izvedejo skupaj. Poznamo vse povezave, ki jih ima komponenta, ki vsebuje tak segment, znotraj segmenta z drugimi komponentami. Takoj po izvedbi tega segmenta izvedemo potrebne teste in v primeru napake razveljavimo dobljene rezultate. Ker poznamo vse povezave z drugimi komponentami in je le-teh omejeno število, je to dokaj preprosto. V končni fazi postavimo vse komponente v stanje, v kateri so bile, tik preden se je pojavila napaka. S tem smo odpravili posledice odpovedi.

3.4 Varnost in zaščita

V vsakem računalniškem sistemu morajo obstajati metode in mehanizmi za zaščito vseh objektov v sistemu. S tem je vsakemu uporabniku zagotovljena varnost. To pomeni, da so podatki in servisi dostopni samo pooblaščenim strežnikom in uporabnikom, da so servisi, ki jih nudijo strežniki, ustrezni, ter da vedno poznamo identiteto uporabnikov servisov. To potrebujemo zaradi nadzora delovanja sistema in analize ob zlorabi in za obračunavanje.

Zaščita objektov pomeni nadzor dosega do le-teh. Objekti v sistemu morajo biti zaščiteni pred nepooblaščenimi uporabniki in pred nenadzorovano uporabo zaradi napak in odpovedi v programski ali strojni opremi. Pri problemih zaščite v računalniških sistemih je treba ugotoviti, komu lahko zaupamo. V centraliziranih sistemih so bili mehanizmi za zaščito zbrani v jedru operacijskega sistema. Jedro je bilo namreč ustrezno zaščiteno pred ostalimi deli sistema in so mu lahko vsi zaupali.

Pri porazdeljenih sistemih sta dva problema, ki narekujeta drugačno reševanje. Ker so jedra operacijskega sistema porazdeljena v sistemu, zelo lahko pride do poskusov spreminjanja le-teh in do nedovoljenih operacij v sistemu. Drugi razlog za

uporabniku.

V teku razvoja sistemov so se načini in možnosti za zagotavljanje teh ciljev spreminjali. Pri prvih računalnikih so bile procesne in pomnilniške zmogljivosti zelo drage. Zato si ni bilo mogoče predstavljati, da bi računalnik sam izvajal kakršnekoli upravljalske funkcije. Uporabnikov je bilo zelo malo in vsak je bil hkrati programer, operater in upravljalec sistema. Sam je poskrbel za nalaganje programov in podatkov, za izvajanje programa in za izpis rezultatov. Ko so se zmogljivosti računalnikov povečevale in se je tudi cena procesiranja zniževala, je bilo uporabnikov vedno več. Računalniki so že imeli enostavne upravljalske mehanizme, ki so se imenovali paketni sistemi. Le-ti so skrbeli za zaporedno nalaganje programov in podatkov, za izvajanje programov in za shranjevanje rezultatov. Vsak program se je izvedel do konca in šele potem je sistem naložil nov program. Tak sistem je bistveno izboljšal uporabo virov v sistemu, saj je bilo mnogo manj izgube časa med posameznimi posli kot prej pri ročnem upravljanju. Naslednja težava, ki je onemogočala boljšo izrabo zmogljivosti, je bila razlika v hitrosti delovanja procesnih enot in vhodno-izhodnih enot. Zato so uvedli tako imenovano istočasno računanje in vhod-izhod, dobro poznano po začetnicah angleškega opisa te dejavnosti kot SPOOLer (simultaneous peripheral operation on line). To je bila prva oblika kvazi-paralelnega izvajanja več dejavnosti, ki je zahtevala določene zmožnosti v strojni opremi. Tu mislimo predvsem na prekinitvene mehanizme. Na ta način je računalnik prevzel skrb za upravljanje mnogih virov in v večji meri omogočil doseganje zgoraj opredeljenih ciljev.

Naslednji korak, ki je računalnik še bolj približal uporabniku, je bil uporaba operacijskih sistemov z dodeljevanjem časa (time sharing). Tak operacijski sistem hkrati več uporabnikom omogoča interaktivno delo. Za razliko od paketnega sistema se tu program praviloma ne izvrši naenkrat. Uporabnik dobi pravico do uporabe virov v sistemu samo za kratke intervale, vendar ima zaradi velike hitrosti delovanja občutek, kot da uporablja računalnik sam. To seveda velja, če sistem ni preobremenjen. Viri v sistemu so dobro izkoriščeni, vendar zahteva režijsko delo pri menjanju dejavnosti precej časa.

Čim boljši so bili računalniki in čim večje so bile zmogljivosti, več uporabnikov jih je uporabljalo in pojavljale so se nove zahteve. Z optimizacijo algoritmov in programov in z novimi tehnološkimi rešitvami so nekaj časa uspeli zadovoljevati zahteve. Bolj dolgoročna rešitev pa je porazdelitev procesiranja in paralelnost.

Prvi problem, ki so se ga lotili strokovnjaki, je bilo upravljanje komunikacij. Prenos podatkov med porazdeljenimi procesnimi mesti mora biti zanesljiv in brez napak. Vsako procesno mesto je imelo svoj lokalni operacijski sistem. Lokalni operacijski sistem uporablja servise komunikacijskega sistema in nudi servise uporabniku. Razen zanesljive komunikacije v teh zgodnjih oblikah porazdeljenih sistemov ni bilo posebnih pripomočkov, ki bi uporabniku pomagali pri delu v porazdeljenem sistemu [Fort86].

Uporabniki pa so potrebovali mehanizme, ki bi jim omogočili bolj celovit pogled na porazdeljen sistem in čimvečjo transparentnost porazdeljenosti sistema. *Mrežni operacijski sistem* je bil nov nivo, ki so ga dodali med komunikacijski sistem in lokalni operacijski sistem na vsakem procesnem mestu. Značilnost mrežnega operacijskega sistema je, da vsak uporabnik še vedno v glavnem dela ne enem procesnem mestu, da se zaveda porazdeljenosti objektov v sistemu in da sistem vsebuje zelo malo mehanizmov za reševanje iz napak [Trip87]. Glavna področja, kjer uporabik čuti porazdeljenost so datotečni sistem, zaščita in izvajanje programov [Tane85]. To pomeni, da mora uporabnik še vedno sam reševati množico problemov, povezanih z upravljanjem sistema.

Porazdeljeni operacijski sistem omogoča uporabniku pogled na porazdeljen sistem kot celoto, brez mej med posameznimi procesnimi mesti. Upravljanje se nanaša na vse objekte v sistemu in optimira rabo vseh virov, za razliko od mrežnega operacijskega sistema, kjer gre za optimizacijo delovanja vsakega posameznega procesnega mesta. Porazdeljen operacijski sistem je v enakem odnosu do porazdeljenega računalniškega sistema, kot lokalen operacijski sistem do centraliziranega računalniškega sistema. Porazdeljen operacijski sistem deluje enotno, vendar ima svoje kopije na vseh procesnih mestih, njegove funkcije pa se izvajajo paralelno in med seboj sodelujejo.

3 Upravljalske funkcije

Pri pregledu upravljalskih funkcij se bomo omejili predvsem na tiste, ki so povezane z operativnim upravljanjem porazdeljenih sistemov in so realizirane v porazdeljenih operacijskih sistemih.

3.1 Upravljanje konfiguracije

Porazdeljen računalniški sistem lahko predstavimo kot množico komponent programske in strojne opreme, ki so lahko prostorsko porazdeljene. Vendar pa opisana množica tvori porazdeljen sistem šele, ko so komponente povezane in sodelujejo pri reševanju problemov. Funkcija upravljanja konfiguracije sistema poskrbi za začetno povezavo komponent in vzpostavitev delovanja sistema in tudi za dinamično prilagajanje sistema, ki je potrebno zaradi napak ali spremenjenih potreb uporabnikov.

Določene operacije pri konfiguraciji so dolgoročnega značaja in jih mora opraviti operater. Mislimo predvsem na instalacijo osnovnih komponent programske in strojne opreme in fizično povezavo posameznih komponent sistema. Ko je najbolj osnovna konfiguracija sistema vzpostavljena, lahko nadzor nad konfiguracijo sistema prevzame ustrezen strežnik. Večina ostalih aktivnosti, ki jih vključuje upravljanje konfiguracije, se izvaja dinamično. Tako se razporejajo aktivnosti sistema glede na trenutno zasedenost posameznih virov v sistemu in njihovih sposobnosti za opravljanje posameznih operacij. To je tudi edini način za optimizacijo zmogljivosti, ki se lahko izvaja dinamično. Ostali načini so dolgoročnega značaja in zato ne sodijo v okvir naše obravnave. Tudi v primeru odpovedi komponent se mora konfiguracija sistema spremeniti tako, da uporabniku ostanejo na razpolago vsi servisi, čeprav se zmogljivost sistema zmanjša. Komponente, ki so bistvene za zagotavljanje določenih servisov, ni pa jih mogoče nadomestiti z drugimi komponentami, morajo biti zato replicirane. Poleg skrbi za povezave med posameznimi komponentami sistema mora upravljalska aktivnost v vsakem trenutku posredovati informacije o stanju sistema, če uporabnik to zahteva.

V splošnem pa je zaželjena transparentnost lokacije virov, torej se uporabnik ne ukvarja z iskanjem lokacije vira. Transparentnost lokacije vira je tesno povezana s poimenovanjem virov in semantično konsistentnostjo strežnikov in pomeni, da je njihov servis enak, ne glede na to, kje v sistemu se strežnik nahaja. Večje težave se pri tem pojavijo v porazdeljenih sistemih, ki ga sestavljajo zelo različne komponente strojne ali programske opreme. Transparentnost lokacije virov lahko dosežemo na različnih nivojih, vendar je najbolje, če je realizirana v jedru operacijskega sistema, saj je tako dostopna najširšemu krogu uporabnikov.

realizacijo mehanizmov zaščite izven jedra pa je želja po čim manjšem jedru.

Za zaščito sta v porazdeljenih sistemih uporabljena dva mehanizma: seznam za nadzor dostopa in mehanizem prenašanja pravic.

Seznam za nadzor dostopa do vira vsebuje vse uporabnike, ki imajo pravico dostopa do tega vira. Seznam je pridružen strežniku, ki vsebuje vir. Zahtevek, s katerim uporabnik zahteva servis, vsebuje tudi enolično oznako uporabnika. Ta del sporočila je tako zaščiten, da ga ni mogoče ponarediti. Dobra lastnost mehanizma s seznamom je tudi to, da je mogoče enostavno razveljaviti pravice v zvezi z dostopom, če je to potrebno. To je potrebno ob odpovedih in pojavih napak, pri spremembah konfiguracije sistema ali zaradi optimizacije zmogljivosti sistema.

Osnovni element drugega mehanizma je posebna vstopnica ali žeton, ki ga imenujemo pravica [Tane84]. Le-ta omogoča svojemu lastniku dostop do virov ali izvajanje določenih operacij, ki so vključene v dani strežnik. Uporabnik, ki želi generirati nov objekt, pošlje zahtevek za to ustreznemu strežniku. Zahtevek vsebuje tudi neko naključno število iz določenega intervala. Strežnik generira nov objekt, hkrati pa zgenerira tudi pravico za dostop do tega objekta. S pomočjo naključnega števila, ki ga je poslal uporabnik, šifrira pravico in jo vrne uporabniku. Le-ta jo lahko potem razmnoži in jo preda tudi drugim strežnikom in uporabnikom. Strežnik shrani naključno število v posebno tabelo, povezano z novim objektom. Ko pride zahtevek za doseg novega objekta, strežnik preveri pristnost pravice s številom, shranjenim v tabeli. Pravica je zaščitena tako, da je ni mogoče ponarediti. Pravice je težko razveljaviti in to je slaba lastnost tega mehanizma. Ko strežnik, ki je odgovoren za podani objekt, generira pravico, jo šifrira z naključnim številom. To pomeni, da uporabnik, ki je zahteval kreiranje novega objekta, sam ne more spremeniti vsebine pravice in omejiti pravice uporabe posameznih operacij nad objektom uporabniku, ki bi mu želel dati kopijo pravice. To stori lahko le tako, da pošlje zahtevek za kreiranje take pravice strežniku.

Opisane lastnosti mehanizma s pravicami omogočajo porazdeljeno realizacijo zaščite. Zaradi porazdeljenosti sistema potrebujemo metode za šifriranje sporočil, ki onemogočajo prisluškovanje na komunikacijskih medijih, in overjanje sporočil s tako imenovanim digitalnim podpisom. Najbolj je razširjeno šifriranje po tako imenovanem DES (Data Encryption Standard) standardu. Oba osebka, ki komunicirata, morata poznati ključ, to je neko dovolj veliko število, ki služi kot osnova za šifriranje in dešifriranje sporočil. Metoda šifriranja je poznana, to pomeni, da je vsa skrivnost v ključu. Funkcije za šifriranje sporočil so realizirane v siliciju in zato je šifriranje v realnem času mogoče tudi za velike hitrosti prenosa podatkov, ki so potrebne v porazdeljenih sistemih. Problem predstavlja samo generacija in distribucija ključev. Zato sta Diffie in Hellman razvila nov sistem, imenovan sistem šifriranja z javnim ključem. Poznana sta sistem šifriranja in dešifriranja in tudi ključ za šifriranje. Ključa za šifriranje in dešifriranje sta pridobljena iz istega osnovnega ključa. Vsa skrivnost sistema je v posebnosti funkcije, s katero iz osnovnega ključa dobimo ključ za šifriranje. Ta funkcija je zelo enostavno izračunljiva, njena inverzna funkcija pa ni izračunljiva v realnem času. Zato ni mogoče učinkovito izračunati osnovnega ključa in iz tega ključa za dešifriranje.

3.5 Obračunavanje

Obračunavanje je sistemska funkcija, ki spremlja in beleži uporabo virov in servisov v sistemu. Poleg sestavljanja računov na osnovi pogodb in tarif za posamezne servise, ta funkcija omogoča tudi omejevanje uporabe virov in servisov in zagotavlja poštenost pri uporabi le-teh. V tesno sklopljenih sistemih, ki so običajno v lasti ene same organizacije in kjer se stroški pogosto ne delijo med uporabnike, prevladujejo predvsem ti drugi motivi.

Tarife za uporabo posameznih servisov se določijo na osnovi stalnih stroškov, spremenljivih stroškov in posebnih stroškov. Stalni stroški se plačujejo periodično in so neodvisni od uporabe sistema. Posebni stroški so odvisni od časa uporabe ali količini obdelanih in prenešenih podatkov, kvalitete servisa in oddaljenosti strežnika. Posebni stroški so povezani s servisi, ki niso vključeni v osnovni nabor [Slom87].

Pravice za uporabo posameznih virov lahko izrazimo z virtualnim denarjem, ki ga ima uporabnik [COST84]. Z virtualnim denarjem si kupi pravice za uporabo virov oz. servisov. Virtualni denar lahko uporabnik dobi na osnovi dogovora med uporabniki ali kot protivrednost denarja, ki ga plača za uporabo sistema. Virtualni denar je v različnih valutah, za vsak vir ali servis druga valuta.

Vse operacije z denarjem so tesno povezane z uporabo mehanizmov za varnost in zaščito.

3.6 Upravljanje z imeni

Imena imajo v računalniških sistemih zelo pomembno vlogo, saj z njihovo pomočjo identificiramo izbrane objekte v sistemu na vseh nivojih abstrakcije. Imena uporabljamo na različnih področjih upravljanja. Še posebno pomembna je vloga imenskega sistema v tesno sklopljenih računalniških sistemih zaradi dovoljene raznolikosti gradnikov sistema in želje, da bi uporabnik videl sistem kot enovito celoto.

Zaradi čimboljše prilagodljivosti sistema želimo, da ime vsebuje čim manj informacij o objektih, saj se lahko le-te dinamično spreminjajo [Terr86]. Te informacije morajo zato biti shranjene neodvisno in organizirane tako, da je spreminjanje in dodajanje podatkov o objektih enostavno, hkrati pa je enostavna tudi povezava z imeni. To namreč omogoča učinkovito delovanje imenskega sistema.

4 Literatura

[COST83] T. Kalin (edt.),
Management issues in Local Area Networks,
COST 11 bis LAN Management Group Report,
Proc. EUTECO '83, North Holland, 1983

[COST84] A. Langsford (edt.),
Distributed Systems Management in Wide Area Networks, report by COST 11 bis DSM Group,
February 1984

[Fort86] P.J. Fortier,
Design of Distributed Operating Systems
McGraw-Hill, New York, 1986

[LeLa81] G. LeLann,
Error recovery, in Distributed systems,
LCNS 105, Springer-Verlag, Berlin 1981

[Slom87] M. Sloman, J. Kramer,
Distributed Systems and Computer Networks,
Prentice-Hall International, 1987

[Tane84] A.S. Tanenbaum et.al.,
Capability Based Protection in Distributed
Operating Systems, Proc. Symp. Certificering
van Software, Utrecht, November 1984

[Tane85] A.S. Tanenbaum, S.J. Mullender,
Distributed Operating Systems,
ACM Computing Surveys, vol.17/4,
December 1985

[Tane87] A.S. Tanenbaum, R. van Renesee,
Reliability Issues in Distributed Operating
Systems, Proc. 6th Symp. Reliability
of Distr. Softw. & Datab. Systems,
Williamsburg, Virginia, March 1987

[Terr86] D.B. Terry,
Structure-free Name Management for
Evolving Distributed Enviroments
IEEE Proc. 6th ICDCS, May 1986

[Trip87] A. Tripathi,
Distributed Operating Systems,
Tutorial no.4, 7.ICDCS,
W.Berlin, September 1987