UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Aleksander Pahor

ANALIZA IN PRENOVA SISTEMA UPRAVLJANJA Z DOKUMENTACIJO V PODJETJU

Diplomsko delo
na visokošolskem strokovnem študiju

Mentor: dr. Mojca Ciglarič

Ljubljana, 2009

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00425/2009
Datum: 15.01.2009

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **ALEKSANDER PAHOR**

Naslov: **ANALIZA IN PRENOVA SISTEMA ZA UPRAVLJANJE Z DOKUMENTACIJO**

DOCUMENT MANAGEMENT SYSTEM: ANALYSIS AND RENOVATION

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija

Tematika naloge:

Predstavite poslovno okolje v podjetju in podrobno analizirajte njihovo upravljanje z dokumentacijo: poimenovanje datotek, dostop do njih, možnosti obnovitve podatkov, ugotavljanje časa, vsebine in avtorja zadnjih sprememb in podobno. Na podlagi ugotovljenih težav identificirajte možna orodja, ki bi jih v podjetju lahko uporabili za izboljšanje stanja. Navedite kriterije za izbiro orodij ter nato izberite ustrezno orodje. Opišite namestitev in postopek uvajanja v podjetje ter izkušnje ob začetku produkcijske uporabe. Kritično ovrednotite rezultat.

Mentor:

M. Ciglarič

doc. dr. Mojca Ciglarič

Fakulteta za računalništvo in informatiko
2
UNIVERZA V LJUBLJANI

Dekan:

prof. dr. Franc Solina

I Z J A V A O A V T O R S T V U

diplomskega dela

Spodaj podpisani/-a _____,

z vpisno številko _____,

sem avtor/-ica diplomskega dela z naslovom:

_____

_____

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal/-a samostojno pod mentorstvom (naziv, ime in priimek)

_____

in somentorstvom (naziv, ime in priimek)

_____

- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela

- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne _____ Podpis avtorja/-ice:_____

Zahvala

V prvi vrsti se zahvaljujem svoji mentorici dr. Mojci Ciglarič za potrpljenje, ki ga je izkazala z menoj. Predvsem cenim to, da je bila pripravljena z nekaterimi izdelki počakati, kar je bilo pogojeno z mojim delom, ki velikokrat ne dopušča, da bi se svojim ostalim obveznostim posvetil toliko, kolikor bi si zaslužile.
Zahvaljujem se vsem v podjetju Hermes Softlab d.d., ki so mi vedno stali ob strani in mi pomagali odrasti strokovno, poslovno in osebno. Davorju Hvali, ki me je vzel v službo in vsem mojim nadrejenim: Alešu Pestotniku, Primožu Svetku, Mihi Urbaniji in Alešu Koširju, ki so mi zaupali vedno bolj odgovorne naloge, ki so mi omogočile videti svet in delati na mnogih projektih in podjetjih. To mi je dalo samozavest, da lahko suvereno nastopim pred komerkoli. Zahvala gre tudi vsem sodelavcem podjetij SEZ Ag. in NLB d.d., s katerimi sem strokovno izredno napredoval.
Zahvaljujem se svojim sošolcem na fakulteti, ki so mi pomagali, da sem vedno dobil vse zapiske in bil seznanjen z vsem, kar se je na fakulteti dogajalo, posebno Gašperju Petraču in Primožu Furlanu, s katerima sem se učil za izpite in brez katerih bi mi bilo veliko težje pri opravljanju le-teh.
Nazadnje, ampak najbolj toplo, se zahvaljujem svoji družini, ki nikoli ni obupala nad tem, da bom zaključil študij. Za vsa priganjanja in opominjanja moji največji zaveznici, prijateljici, punci in sedaj ženi Vesni. Mojim staršem Boji in Zdravku, ki sta me vzgojila z vso ljubeznijo in neumorno prenašala vsa zavlačevanja pri študiju. Babi in đedu za vedno toplo besedo in brezpogojno podporo pri vsem, kar sem in bom počel. Mojemu mlajšemu bratu Klemnu in moji hčerki Živi pa za vseprisoten opomin, da je potrebno biti včasih komu tudi v zgled.

Vesni

Kazalo

Prevodi in razlage kratic in tujk

ACL	Access Control List	Seznam nadzora dostopov
API	Application Programming Interface	Vmesnik za programiranje aplikacij
ASCII	American Standard Code for Information Interchange	Ameriške standarne kode za izmenjavo informacij
CAU		Centralna Administracija Uporabnikov
	Changeset	Množica sprememb je nedeljiv paket spremenjenih datotek
CM	Configuration managment	Upravljanje z nastavitvami
CMS	Content Management Systems	Sistem za upravljanje z vsebinami
CVS	Concurrent Version Control System	Sistem za nadzor hkratnih različic omogoča večjemu številu uporabnikom istočasno nadzorovan in beležen dostop do datotek v skupni rabi
	Datamining	Podatkovno rudarjenje. Opisuje dejavnost iskanja podatkov po bazah podatkov ali drugih oblikah zbirk in zapisov podatkov
EOL	End of line	ASCII koda znaka za konec vrstice
EVS		Naslednik CVSNT strežnika
EXT	EXTender file system	Tip datotečnega sistema
	Floating licence	Konkurenčna licenca – omogoča delo z eno licenco več različnim uporabnikom, vendar samo enemu od njih naenkrat
Globus		Aplikacija za transakcijsko poslovanje s pravnimi osebami
GNU/GPL	GNU General Public Licence	Tip licenciranja, ki omogoča odprtokodne rešitve
GSSAPI	Generic Security Service API	Splošni vmesnik za varnostne storitve
GUI	Graphic User Interface	Grafični uporabniški vmesnik
	High-end	Navadno opisuje nekaj, kar je objektivno najboljše v svoji skupini ali področju
IDE	Integrated Development Enviroment	Integrirano razvojno okolje
IIS	Internet Information Server	Strežnik spletnih strani
IT	Information Technology	Informacijske Tehnologije
KIT	Katalog Informacijske Tehnologije	Zbirka obrazcev in predpisov informacijske tehnologije podjetja NLB d.d.
	Memory leak	Dogodek, ki se zgodi, ko se v aplikaciji neka podatkovna struktura ne sprošča dovolj skrbno in ostanejo deli pomnilnika zasedeni, kljub neuporabi, skozi čas pa se velikost tako zasedenega pomnilnika veča
MSI	Microsoft Installer	Aplikacija za nameščanje, odstranjevanje in spreminjanje aplikacij v operacijskih sistemih Windows (.msi datoteka)
	Multisite	Opisuje geografsko razpršene sisteme, storitve ipd.
Mvfs	MultiVersion FileSystem	Tip datotečnega sistema, ki omogoča shranjevanje različic
NBO		Novo Bančno Okence. Aplikacija za uporabnike po poslovalnicah, ki združuje različne zaledne sisteme.

NDCS		Sistem za nadzor različic elementov sistema Globus
NLB d.d.		Nova Ljubljanska Banka delniška družba
NTFS	New Technology File System	Standardni datotečni sistem Windows NT družine operacijskih sistemov Microsoft
OS	Operating System	Operacijski Sistem
PC Team		Skupina znotraj UCIT, ki skrbi za namestitve osebnih računalnikov
PSERVER		Avtentikacijski protokol
	Product	Izdelek ali večinoma pogovorno produkt opisuje navadno aplikacijo v širšem pomenu besede
RC	Revision Control	Nadzor različic
RCS	Revision Control System	Sistem za nadzor različic
Revizija	Verzija	Različica
	root	Koren
SCCS	Source Code Control System	Sistem za nadzor izvorne kode
SCM	Source Code Management	Sistem za nadzor različic izvorne kode
SMS	Microsoft Systems Management Server	Sistem za upravljanje sistemov Microsoft
SSH	Secure SHell	Varna lupina
SSP	Security Service Provider	Ponudnik storitve varnosti
SSPI	Security Support Provider Interface	Vmesnik storitve podpore varnosti
SVN	SubVersion	Kratica za aplikacijo SubVersion
UCIT		Upravljalski center informacijske tehnologije
UNIX		Družina operacijskih sitemov
UTF	Unicode Transformation Format	Format zapisa nabora znakov
VCS	Version Control System	Sistem za nadzor različic
VPN	Virtual Private Network	Navidezno zasebno omrežje
VSS	Visual Source Safe	Kratica za aplikacijo Visual Source Safe
	Wrapper	Opisuje dejavnost programske opreme, da deluje kot adapter med programskimi rešitvami

Pojmi CVSNT

»***Različica ali revizija ali verzija***« (*Revision, Version*) je stanje neke datoteke ob točno določenem času. *Različica* ima neko interno številko, ki je odvisna od zaporednega števila in morebitnega *razvejanja*. Številka različice je interna in se je ne da prirejati. Vse različice so vrste 1.n ali 1.n.m, vrednost ne bo nikoli dosegla 2.0. Vsaka *različica* ima lahko prilepljeno eno ali več *oznak,* s katerimi bolje nadzorujemo, kaj je vsebina *različice,* in seveda *komentar avtorja* in *čas objave*.

»***Skladišče***« (*Repository*) je osnovno skladišče kjer so shranjene datoteke pod nadzorom hkratnih različic. Skladišče se nahaja na strežniški strani. Uporabnik mora poznati samo ime njegovega korena (*root*). Navadno (tudi priporočeno) potrebujemo samo eno *skladišče*, saj lahko v njem ustvarjamo več *modulov*.

»***Veja***« (*Branch*). Zaradi lažje predstave različice organiziramo tako, da so bolj pregledne. *Veja* navadno pomeni, da bo neka skupina ali uporabnik dalj časa spreminjal datoteko in zato ne želi motiti ostalega razvoja, ki se nadaljuje na glavni *veji*. Po končanem razvoju na *veji* je potrebno *spojiti* razvojno vejo v glavno vejo.

»***Oznaka***« (*Tag*) Na *različico* lahko nalepimo napis ali oznako, s katerim bomo kasneje lahko operirali. Ob *prevzemu* lahko podamo oznako in tako pridobimo vse *različice* datotek, ki imajo to oznako na sebi. O*znako* uporabljamo za bolj močne in širše opombe, ki zadevajo večje število datotek – pa tudi za nadzor na *izdajami*.

»***Izdaja***« (*Release*) je skupek datotek, ki so označene s skupno *oznako*, za katero smo se odločili, da bo *oznaka* izdaje. *Izdaja* je navadno nek cilj v funkcionalnostih ali časovnem obdobju, pri kateri se ustavimo z razvojem in pogledamo celotno sliko stanja datotek našega izdelka. Predvsem je uporabno, da se lahko (če smo pravilno in celotno označili vse datoteke) kadarkoli vrnemo na določeno *izdajo* tako, da podamo *oznako izdaje* ob *prevzemu* datotek.

»***Modul***« (*Module*) je ena od osnovnih map skladišča. Navadno predstavlja zaključeno skupino datotek. Teoretično lahko imamo tudi samo en *modul* in v njem navadne mape, je pa bolj praktično (če je možno) natančno razmejiti datoteke po skupinah, ki jih bodo uporabljale oz. po namenu shranjenih datotek. Ob *prevzemu* je potrebno izbrati tudi *modul* in tako ustvariti lokalno kopijo *modula* v svojem *peskovniku*.

»***Peskovnik***« (*Sandbox*) je prostor na lokalnem disku, kjer se nahajajo *prevzete* datoteke iz *skladišča*. Iz *peskovnika* datoteke pošljemo na strežnik tako, da jih *objavimo*. Datoteka, ki se nahaja v *peskovniku* in še ni *objavljena*, še ni dostopna ostalim uporabnikom ampak samo nam.

»***Prevzem***« (*Checkout*) sproži kopiranje datotek iz določenega *modula skladišča* pod nadzorom strežnika CVSNT na lokalni disk. Datoteke se s strežnika prenesejo v *peskovnik*, kjer so nam na razpolago za popravljanje. Imamo možnost izbire po datumu, *reviziji* ali *oznaki*. To nam omogoča, da imamo lokalne datoteke iz *modula* na izbrani dan ali vse datoteke, ki so označene z določeno *oznako*. Vsak prevzem sproži kopiranje samo ene različice datoteke. Brez povezave s strežnikom ni mogoče *pregledovati zgodovine*, *grafa*

revizij, *razlik* med *različicami* datoteke itd. Lahko pa nemoteno urejamo *prevzeto različico* datoteke.

»***Objavi***« (Commit) je funkcija, s katero se sprememba na datoteki shrani na CVSNT strežnik. Samodejno se ji dodeli nova številka *različice*. Sedaj je ta *različica* dostopna tudi vsem ostalim uporabnikom, če sprožijo *posodobitev* svojega *peskovnika*. Objavimo lahko več datotek hkrati ali celotno mapo. Ne moremo pa objaviti različice, ki je enaka predhodni. Če je datoteka binarna (.doc, .avi, .xls, ...), se shrani celotna datoteka še enkrat; če je datoteka tekstovna pa samo razlika med prejšnjo različico. Metoda s shranjevanjem samo razlik (t.i. delta) binarne datoteke v CVSNT ne deluje vedno v redu in še v eksperimentalni fazi.

»***Izdaj***« (*Release*) pobriše izbrane datoteke/mape iz *peskovnika*. Vse ostaja še vedno shranjeno na strežniški strani. Lahko izbiramo med tremi načini *izdaje*.
- Brisanje celotnega *peskovnika* brez spremenjenih datotek in datotek, ki niso pod nadzorom CVSNT.
- Brisanje popolnoma vseh datotek.
- Brisanje kontrolnih datotek CVSNT (rekurzivno).

Zadnji način je uporaben za spremembo *peskovnika* v navadno mapo. Taka mapa ni več povezana s CVSNT.

»***Odstrani***« (*Remove*) izbriše datoteko/mapo na strežniški strani. Datoteka ali mapa se pri tem z *zgodovino* vred izgubita za vedno. Uporaba tega ukaza je odsvetovana, saj dejansko ni potrebe po trajni *odstranitvi* datoteke in njene celotne *zgodovine*. Administrator lahko, preko zaporedja ukazov za oživitev, ponovno povrne tudi izbrisano datoteko.

»***Posodobi***« (*Update*) posodobi peskovnik z zadnjimi spremembami, ki so že na strežniku (kopirajo se samo nove datoteke – datoteke, ki so trenutno urejane ali spreminjane ostanejo nedotaknjene). *Posodobitev* je sprožena z istimi kriteriji, kot je bilo sproženo *prevzemanje* (*datum*, *različica* ali *oznaka*). S posebno posodobitvijo lahko spremenimo tudi te kriterije.

»***Dodaj***« (*Add*) doda izbrane datoteke/mape na CVSNT strežnik. Sedaj je ta datoteka dostopna tudi vsem, če sprožijo *posodobitev* svojega *peskovnika*. Ta datoteka je sedaj v skladišču. Po *dodajanju* moramo vedno tudi *objaviti* datoteko ali mapo. Dodaj torej le rezervira ime za prvo objavo.

»***Prezri***« (*Ignore*) izbrane datoteke/mape označi za ignoriranje. Take datoteke ne bodo *objavljene* v *skladišču*, če sprožimo *objavljanje* nad mapo. *Prezrte* datoteke/mape so dostopne samo v lastnem *peskovniku*.

»***Sledi***« (*Watch*) nad izbrano datoteko/mapo je način, s katerim vsak uporabnik zaklene datoteko tako, da bo vedno obveščen preko elektronske pošte, ko jo bo nekdo popravljal. Katerikoli uporabnik lahko sproži »*Uredi*« in jo popravi. Ukaz se lahko sproži samo iz ukazne vrstice. Z istim ukazom tudi umaknemo sledenje z datoteke.

»***Uredi***« (*Edit*) je ukaz, s katerim nam je dovoljeno popravljati datoteko, ki je *sledena*. Po tem ukazu je datoteka pisljiva. Kdor *sledi* datoteko, bo po elektronski pošti obveščen, da smo odstranili zaklep na datoteki.

»***Razveljavi urejanje***« (*Unedit*) je ukaz, s katerim prekličemo popravljanje datoteke, ki je sledena. Po tem ukazu je datoteka znova nepisljiva. Uporabnik, ki je datoteko *sledil*, bo obveščen po elektronski pošti, da je bilo *urejanje* datoteke razveljavljeno.

»***Razlikuj***« (*Diff*) prikaže razlike do delovne različice. Če držimo CTRL tipko med izbiranjem, lahko podrobneje določimo izbiro *različice*.

»***Razloži***« (*Annotate*) je ukaz, s katerim lahko v tekstovnih datotekah za vsako vrstico vidimo, kdo in v kateri *različici* jo je dodal.

»***Osveži stanje map***« (*Refresh folder status*) ponovno prebere celotno mapo in ponovno izriše ikone nad datotekami. Priročen ukaz, s katerim preverimo stanje datotek, če domnevamo, da ikone niso pravilne oz. ažurne. Pomembno je razlikovati med tem ukazom in ukazom *posodobi*. *Osveževanje stanja map* ne prenaša datotek in *posodablja peskovnika*, ampak samo preveri ikone nad datotekami.

»***Graf revizij***« (*Revision graph*) nam slikovno prikaže *drevo* zgodovine *različic* datoteke. V takem pogledu natančno vidimo vse *komentarje* ob *objavi*, vse *veje*, *oznake* in *spajanja*. Zelo uporabna funkcionalnost, s katero lahko s klikom na desni gumb določamo tudi, katero različico želimo (*lepljiva različica*). *Različica*, ki jo imamo trenutno v svojem *peskovniku*, je prikazana odebeljeno in jo tako lažje opazimo.

»***Zgodovina***« (*History*) je dostopna v večih področjih TortoiseCVS. *Zgodovina* je bistvo *sistema nadzora različic*. *Zgodovina* dostopna na desnem kliku, je tabelarni izpis dogodkov nad datoteko. Prikazuje enake podatke kot *graf revizij*, vendar je drugačnega videza in je zato morda lažji način za pregledovanje komentarjev (težje vidna pa so *spajanja* in *oznake*).

»***HEAD veja***« (*HEAD branch*) je glavna *veja* v *grafu revizij*. Navadno *spajamo* vse *veje* v *HEAD vejo*. Privzeto *prevzamemo* vedno zadnjo *različico* v *HEAD veji*.

»***Razvejaj***« (*Branch*) je ukaz, s katerim iz trenutne delovne različice ustvarimo vejo, ki jo poljubno poimenujemo.

»***Spoji***« (*Merge*) je ukaz, s katerim spremembe med navedenimi *različicami/oznakami/vejami spojimo* v trenutne delovne vire. *Skladišče* ostane nespremenjeno do naslednje *objave*. CVSNT nima nedeljivega ukaza za spajanje.

»***Lepljiva***« (*Sticky*) pove, da želimo izbrano (na *grafu revizij*) *različico* datoteke od sedaj naprej imeti v svojem peskovniku. Prikazana datoteka v mapi bo od sedaj naprej (do naslednje *posodobitve/prevzema*) starejša od zadnje *različice*. To lahko povzroči, da ne moremo objaviti ničesar kar spremenimo na tako pridobljeni datoteki, saj naslednja različica že obstaja. Za spreminjanje vsebine in kasnejše objavljanje na taki datoteki, moramo nujno ustvariti vejo iz nje in jo kasneje spojiti z zadnjo različico v *HEAD veji*.

»***Uvozi***« (*Import*) je funkcionalnost, s pomočjo katere lahko naenkrat *dodamo* in *objavimo* večje število datotek. Datotekam se takoj dodeli *oznako* in *številko različice*.

POVZETEK

Namen diplomske naloge je predstaviti prenovo nastajanja tehnične in poslovne dokumentacije pri dveh projektih v podjetju NLB d.d. in podati rešitev za bolj učinkovito spremljanje dela na tem področju razvoja in nabiranju poslovnega znanja. Cilj diplomske naloge je dokazati, da sistem za nadzor različic lahko kvalitetno pripomore k večji preglednosti, nadzoru in spremljanju nastajanja spremne dokumentacije pri projektih. Domnevamo, da bo izbrana odprtokodna rešitev, ki omogoča preprosto nadgradljivost, kasnejše dodajanje novih funkcionalnosti po lastni presoji in transparentnost ter ne predstavlja velikega stroška v obliki licenc in zunanjih svetovalcev. Diplomska naloga analizira trenutno stanje in potrebe znotraj dveh različnih projektov podjetja NLB d.d.. Razloženi so poslovni in informacijski vidiki okolja, kjer bo vzpostavljen nov sistem za nadzor različic. Opisan je star način izdelave in hranjenja dokumentacije pri obeh izbranih projektih. Na podlagi potreb in težav analizira, s katerimi kriteriji je mogoče izbrati in vzpostaviti boljši sistem. Na podlagi teh kriterijev so ovrednotene različne izbire skupin ali posamičnih orodij . Iz celotnega nabora aplikacij in orodij, ki so na voljo, se sistematično izbere zelo ozek krog kandidatov, ki se jih nato še bolj podrobno primerja med seboj. V skladu z domnevami se v zaključku postopka izbire odločamo med orodjama SubVersion in CVSNT. Po določitvi izbrane aplikacije CVSNT in posledično spremnih orodij, je opisan natančen postopek in določen obseg spremne dokumentacije, ki je potrebna za odobritev s strani vodilnih v podjetju, da se sistem lahko uvede v obstoječe informacijsko okolje. Na koncu je opisan potek prenove sistema z izbrano aplikacijo za nadzor različic in način uvajanja zaposlenih. V sklepu je opisano zadovoljstvo zaposlenih, uporabnikov in vodilnih z novim sistemom za nadzor različic in podano osebno mnenje za nadaljnje delo na tem področju znotraj celotnega informacijskega sistema podjetja NLB d.d..

Ključne besede:

- nadzor različic, upravljanje z dokumentacijo podjetja, uvajanje sistema v podjetje
- različica, skladišče, peskovnik, oznaka, veja
- CVSNT
- TortoiseCVS

The purpose of this thesis is to present the renewal of the technical and business documentation creation on two projects in the company NLB d.d. and to provide solutions for more effective monitoring of the work and progress of the development and archiving of business knowledge in this segment. The goal is to demonstrate that the system for version control may contribute to greater transparency, control and monitoring of the creation of supporting documentation on projects. We are assuming the choice of an open-source solution which will allow easier upgrading, adding new functionalities later-on with great transparency and will not represent a large cost in the form of licensing and external consulting. In this thesis the current situation is analysed trough the needs of two chosen projects in the company NLB d.d.. Business and IT aspects of the environment where the new version control system will be installed are explained. The old method of creation and storage of documentation on the two selected projects is described, too. Based on the needs and problems it is analysed which criteria is needed to select and establish a better system for version control. Different choices of groups or individual tools are based on those criteria. From the full set of applications and tools that are available on the market a very narrow circle of candidates is than systematically compared in grater detail. As assumed it ends in deciding between applications Subversion and CVSNT. After the selected application, CVSNT, and consequently the supporting tools are defined the detailed process of installation and the volume of supporting documentation is described. The system can be added to the existing IT environment only if the process and documentation comply to the company's standard. Finally, the course of the renovation of the documentation system is described and the method how to teach the employees the new system. Employees, users and managers satisfaction with such new system is described and the personal opinion for further work in this area across the entire company information system is expressed at the end of this thesis.

Keywords:

- version control, enterprise documentation management, introduction of a new system in a company
- version, repository, label, branch
- CVSNT
- TortoiseCVS

1 POSLOVNO OKOLJE IN OPIS SISTEMA PRED PRENOVO

1.1 Namen in izhodišča

V podjetju NLB d.d. sem bil zadolžen za vpeljavo sistema za verzioniranje dokumentacije. Sistem za nadzor izvorne kode in na splošno za upravljanje vseh datotek je nujen za nadzor nad spremembami, do katerih prihaja med razvojem programskih projektov ali dokumentacije. Razvijalci (v širšem pomenu besede – tako tehnologi kot programerji) potrebujejo popolno zgodovino sprememb, da se lahko v primeru kakršnihkoli težav vrnejo k prejšnjim različicam. Novi člani projekta zgodovino lahko pregledujejo kot zbirko znanja in vidijo, kakšna je tipična pot do končnega izdelka. Ker pisanje programov in dokumentacije zahteva veliko časa in denarja, je zelo pomembno porabiti vsaj nekaj časa in denarja tudi za varovanje delnih izdelkov ali izdelkov, ki jih trenutno ne obravnavamo kot uporabne.

Današnja podjetja so vedno bolj odvisna od usklajenosti posameznih članov na različnih projektih. Obsežno znanje, ki ga znamo upravljati in pravilno uporabiti, ustvarja veliko konkurenčno prednost podjetij in posameznikov. V dodatno pomoč pa so nam lahko razne programske rešitve [1]. Jasne smernice in pravila, kako člani projektov dodajajo novo znanje in kako naj ga potem iščejo, niso vedno plod izključno tehnične rešitve. V primeru NLB in podobnih podjetij, ki so na trgu že več desetletij, je upravljanje z nastavitvami, produkt križanja nekega starega, zgodovinskega načina shranjevanja in zadnjih rešitev ter orodij, ki so na voljo. Najboljše rešitve so tiste, ki dobro analizirajo staro stanje, preverijo dodatne zahteve uporabnikov in na podlagi tega ponudijo novo rešitev. S tem dosežejo, da bodo uporabniki sprejeli novo rešitev s kar najmanj odpora. Sodobni zaposleni dobro vedo, da je računalniških rešitev ogromno in ne sprejmejo več česarkoli. Mnogi prihajajo iz različnih podjetjih in projektov, kjer je morda v uporabi že kakšen podoben sistem. Idealno je, da so uporabniki vključeni že v času izbire in implementacije novega sistema. Sprememba okolja in sistema se mora za uporabnike zgoditi neopazno. Sodoben način vodenja projektov skorajda ne dopušča prehodnih obdobij, pri katerih zaposleni delujejo z zmanjšano učinkovitostjo ali z omejenim naborom orodij, aplikacij in pripomočkov. Sistem mora biti robusten in nadgradljiv, predvsem pa transparenten pri svojem delovanju. Po končani vpeljavi morajo ostale spremne službe imeti možnost nadzorovanja področja sistema, ki jih zadeva. To pomeni, da je potrebno izdelati jasna navodila za vsako od teh služb in morebiti nov sistem tudi integrirati v kateri drugi zaledni sistem.

V diplomski nalogi se želim osredotočiti tako na tehnične kot sociološke vidike vpeljave novega sistema v poslovno okolje. Začel bom z analizo starega sistema, ki je bil opravljen preko pogovorov in sestankov z različnimi člani projekta Globus in NBO. Po analizi trenutnega stanja se bom osredotočil na izbiro novega sistema verzioniranja in izbiro orodij, ki bodo nudile celovito rešitev. Na kratko bom opisal način vpeljave v NLB d.d. in oddelke, s katerimi se je potrebno povezati za vzpostavitev sistema v zelo varovanem in dorečenem bančnem okolju.

1.2 Predstavitev poslovnega okolja in trenutnega načina verzioniranja

Bančništvo je stara branža, ki je v drugi polovici 20-ega stoletja izjemno napredovala ravno s pomočjo informacijske tehnologije. Računalniki so v uporabi že od poznih 50-ih let in precej aplikacij, ki so še danes v uporabi, je bilo razvitih že v tistih časih [22]. Nadgradnje, kompatibilnost za nazaj in integracije so v takem okolju dobro poznani pojmi. Obvladovanje tveganj v povezavi z njimi pa strategija, ki jo je ta branža izdelala skozi leta praktičnega dela. Ravno zaradi zrelosti in zavedanja vseh posledic slehernih sprememb je tako okolje težavno za vpeljavo novih sistemov.
Vsako novo aplikacijo je potrebno dobro preučiti z več vidikov:

- dodana vrednost v obliki pohitritve ali lažjega dela uporabnikov
- uporabnost
- težavnost uporabljanja/intuitivnost
- stabilnost
- vzdrževanje na kratki, srednji in dolgi rok
- možnost nadgrajevanja
- možnost integriranja
- varnostni vidiki (belo in črno hackanje)
- cena licenciranja
- spremna dokumentacija
- skupna cena na časovno obdobje

Za kvalitetno oceno zgornjih kriterijev je potrebno dobro poznavanje tako informacijskega kot poslovnega okolja. Ti so ključnega pomena za uspešno vpeljavo dodatnega sistema in s tem novega principa delovanja v poslovno okolje. Selektivnost pri tem poznavanju pa vsekakor ni nič manj pomembna, saj je pogoj za zadovoljivo hitro vpeljavo. Večinoma se računalničarji osredotočamo na tehnične prepreke problemov. Vendar je za dosego cilja neizbežno poznavanje vsaj okvirnega delovanja projektnega vodenja v podjetju, poznavanje sistema organiziranosti podjetja v področjih potrebnih za dosego tega cilja in poslovni delovni okviri uporabnikov - zaposlenih. Prvotno rešujemo težavo ali lajšamo delo poslovnim uporabnikom in to nam mora biti vodilo pri vsakem prepričevanju in argumentiranju. Dobro poznavanje IT sistema je v tem primeru drugotnega pomena za ljudi, ki jim nov sistem predstavljamo. Žal pa je večina dela ravno na tem področju in je zato ostalim soudeležencem take vpeljave nevidna oz. samoumevna. Iskanje kompromisa med tehničnimi možnostmi v že zgoraj naštetih kriterijih in poslovnimi zahtevami je srž uspeha zastavljene naloge.

Na tem mestu je potrebna še krajša razlaga delovnega razmerja, v katerem sem bil prisoten v NLB d.d., saj je pomembna za razumevanje samoomejitve raziskovanja. V podjetju NLB d.d. sem bil prisoten kot delavec podjetja Hermes Softlab d.d.. Podjetje NLB je od nas odjemalo nekatere tehnične storitve, zaradi katerih je bila na njihovi lokaciji poleg mene potrebna nenehna prisotnost še dveh inženirjev. Banka je od nas pričakovala, da opravljamo izključno tiste storitve, zaradi katerih smo bili najeti. Za opravljanje le-teh dejansko ni bilo potrebno natančno poznavanje ničesar drugega kot tehnične rešitve, ki smo jo ponujali banki. Vendar je bilo zaradi razlogov, ki sem jih opisal že na začetku tega poglavja, vseeno potrebno raziskati tako informacijsko, kot poslovno okolje, kjer sem deloval. V podjetju

NLB d.d. sem bil torej gost. Ob nastopu dela sem podpisal izjavo o varovanju podatkov, ki me tudi v bodoče zavezuje k varovanju le-teh. Tudi pisanje tega diplomskega dela je moralo biti odobreno s strani odgovornih v podjetju. Moje dostopne pravice, geslo in nasploh dostop do prostorov so bili ponovno dodeljeni vsake tri mesece. Do zasebnih podatkov nisem imel dostopa. Raziskal sem le najbolj nujne stvari, ki so mi omogočile, da sem svojo nalogo dobro opravil.

1.3 Različna organizacijska okolja

1.3.1 Poslovno okolje – Globus in NBO

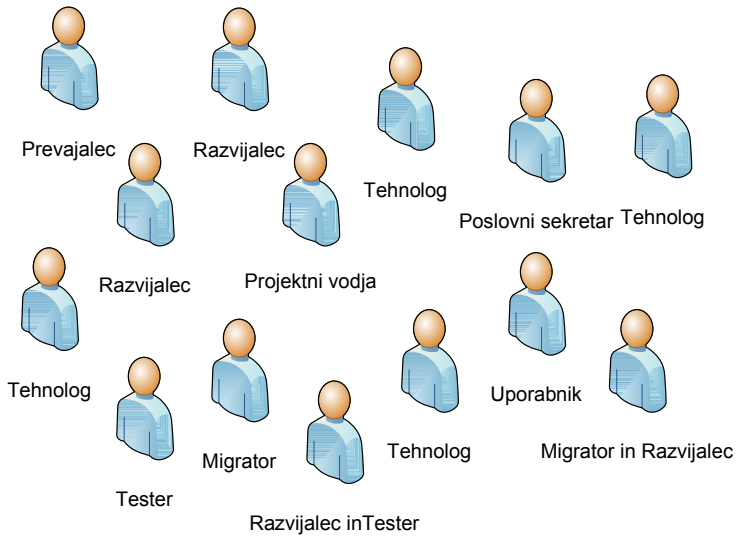
Najbolje je pričeti z grobo razdelitvijo poslovnega okolja banke. Žal je ureditev in razdelitev pristojnosti od oddelka do oddelka in od projekta do projekta različna, vendar obstaja rdeča nit pri vseh tistih okoljih, ki so se me zadevala med to nalogo.

Za sam sistem verzioniranja dokumentacije je najbolj pomembno to, kako se izdelki razvijajo, testirajo in migrirajo med okolji ter kako so razdeljene naloge različnih članov projekta. Oba projekta, pri katerih je bila izvedena prenova sistema dokumentacije, sta razvojna projekta. Globus je razvijal aplikacijo za transakcijsko poslovanje pravnih oseb. NBO pa je »Novo bančno okence« in integrira različne zaledne sisteme, ki jih v poslovalnicah uporabljajo blagajniki, vodje poslovalnic in ostali zaposleni. Gre torej za dve aplikaciji. Vsaka od njiju je razvita v drugačnem programskem jeziku. Verzioniranje izvorne kode poteka v ClearCase za NBO in v lastni aplikaciji za projekt in aplikacijo Globus. Ta nima izvorne kode v pravem pomenu besede ločene v datotekah, zato potrebuje namensko verzioniranje - aplikacijo NDCS, ki verzionira elemente aplikacije Globus. Ti elementi so postavljeni v bazi, preko katere aplikacija deluje. Elementi se v realnem času prevajajo in prikazujejo uporabnikom v obliki aplikacije.

1.3.1.1 Vloge članov projekta

Člani projektov opravljajo različne funkcije znotraj projekta. Te so:

- projektni vodja
- migrator
- tester
- uporabnik
- razvijalec
- tehnolog
- poslovni sekretar / tajnica
- prevajalec



Slika 1.1: Različni člani projekta

Posamezni ključni člani opravljajo tudi do tri različne funkcije (slika 1.1). Oba projekta imata zelo podobno razdeljene vloge članov.

Poslovni sekretarji skrbijo za spremne dejavnosti projekta. Beležijo odsotnosti, sprejemajo korespondenco, rezervirajo sejne sobe in posebno opremo ter posledično dostopajo do področja dokumentacije, ki je potrebna za vodenje zgoraj naštetega (npr. tabele rezervacij, odsotnosti ...). Določene dokumente pa tudi skenirajo in shranjujejo na omrežne vire.

Prevajalci prevajajo dokumentacijo, ki je potrebna neslovenskim članom pri projektu. To so zunanji svetovalci in programerji specialisti. Njihove dokumente ravno tako prevajajo v slovenščino za ostale člane projekta. Navadno dostopajo do dokumentacije vodstva projekta.

Testerji in **uporabniki** v praktičnem smislu opravljajo isto funkcijo. Razlike med njimi so, da so testerji prisotni na lokaciji vodstva projekta, medtem ko so uporabniki v poslovalnicah, in da testerji preizkušajo aplikacijo v življenjskem ciklu razvoja takoj za razvijalci, uporabniki pa za njimi v vmesni večji izdaji aplikacije. Z vidika dokumentacije testerji dostopajo do večje količine dokumentacije in izdelujejo plane testiranj, do katerih imajo kasneje dostop še ostali člani projekta, tudi uporabniki.

Tehnologi ustvarijo največ dokumentacije, saj specificirajo poslovne zahteve, na podlagi katerih razvijalci kasneje razvijajo aplikacijo. Tehnologi pretežno ustvarjajo dokumente, do katerih kasneje dostopajo še ostali člani projekta (razvijalci, testerji, prevajalci, projektni vodje)

Projektni vodje pregledujejo dokumentacijo, ki nastaja pri vseh ostalih. Sami ne ustvarijo veliko dokumentacije, morajo pa imeti dostop do vsega in do metod, s katerimi lahko pregledujejo, kdo in kdaj je ustvaril nov dokument. Projektni vodje odobrijo tehnično dokumentacijo, ki jo proizvedejo tehnologi in nadzirajo napredek in načrte testiranja preko zapisov in planov testiranja.

Razvijalci večinoma ne ustvarjajo dokumentacije. Najpogosteje pregledujejo zahtevke funkcionalnosti, ki jih ustvarijo tehnologi. Izjemoma lahko v primeru napak pregledujejo zapise testnih scenarijev, s katerimi lažje reproducirajo hrošče.

Migratorji z vidika dokumentacije nimajo posebne vloge. Njihovo delo je migriranje zadnjih razvojnih popravkov v okolja za testiranje ali produkcijsko okolje. Potrebujejo pa dostope do migracijskih form, ki so samodejno ustvarjene z orodji za določanje migriranja izvorne kode.

1.3.1.2 Življenjski cikel projektov

Dokumentacija skozi življenjski cikel projekta nastaja različno. Obstaja tudi več različnih življenjskih ciklov glede na različne vrste zahtevkov, ki so predmet projekta.
Različni tipi zahtevkov so:

- poslovne zahteve
- zahteve po spremembah
- napake
- zahtevki za operacije

V primeru poslovnih zahtevkov je prvi dokument, ki nastane *tehnična specifikacija*. Na njegovi podlagi poteka razvoj, ki ob svojem koncu zahteva *zapis o testiranju*. Ob prehodu te funkcionalnosti v naslednje okolje je potrebno sprožiti nastanek *migracijske forme.*

Zahteve po spremembah so v srži podobne novemu *poslovnemu zahtevku*. Vse je enako, le da ustvarimo novo tehnično specifikacijo in po končanem razvoju dobimo nov *zapis o testiranju* ter sprožimo nastanek nove *migracijske forme*.

Zahtevki za operacije so zahtevki, ki razen primerno izpolnjenega obrazca za IT službe, ki zahtevek izvedejo, ne proizvedejo nobene druge dokumentacije.

Napake in opisi napak se beležijo v posebnem sistemu za vodenje zahtevkov. V primeru napake tehnologa ali nedorečenosti, se popravi dokument *tehnična specifikacija* in se po odpravi težave ustvari nov *zapis o testiranju* in nova *migracijska forma*.

1.3.2 IT okolje NLB d.d.

Podjetje NLB d.d. ima zelo dobro razdelano informacijsko okolje. Vzpostavljeni so centri, ki skrbijo za točno določeno področje okolja. Pristojnosti se ne prekrivajo in vsak poseg v okolje je v celotnem življenjskem ciklu dobro dokumentiran.

Krovni center, ki skrbi za vso informacijsko tehnologijo, je UCIT. Center skrbi za vso računalniško podporo poslovnih ciljev bančne skupine. Njegova primarna skrb je obvladovanje produkcije in zahtev različnih področij znotraj banke po informacijski tehnologiji. UCIT sestoji iz treh sektorjev: za obvladovanje produkcije in infrastrukturo, za razvoj in za podporne dejavnosti. V sektor za razvoj spadajo med drugim programski vodje, IT projektni vodje, analitiki, načrtovalci programske opreme in programerji. V sektorju za infrastrukturo je PC Team, v sektorju za podporne dejavnosti pa so CAU, projektna pisarna, trenerji za izobraževanje uporabnikov ter finančni kontroling.

Na kratko bom razdelal, kakšna je funkcija vsakega od sektorjev:

1.3.2.1 Sektor za razvoj

Sektor sestoji iz več področij - programov, ki jih vodijo programski oz. linijski vodje. Programi so namenjeni za podporo naslednjim področjem:

- Elektronski kanali
- Upravljalski sistemi
- Fizične osebe
- Pravne osebe
- Plačilni sistemi
- Finančni trgi

V okviru le-teh se sprejemajo vse velike odločitve o računalniških prenovah in se sledi rokom izdelav in potrošenim virom. Tu se vodijo vsi razvojni projekti in se vzdržuje obstoječe produkcijske kode.

1.3.2.2 Sektor za infrastrukturo

Ta sektor skrbi za nemoteno delovanje produkcije in za vse operativne stvari okoli informacijskih tehnologij. Zaposluje sistemske administratorje in specialiste iz različnih področij računalništva. Večji del specialistov je certificiranih od različnih proizvajalcev strojne ali programske opreme (MSCE, Cisco, Oracle ...). Področja delovanja pokrivajo:

- namestitev, razvoj, nadzor in vzdrževanje delovnih postaj
- namestitev, razvoj, nadzor in vzdrževanje strežnikov (spletnih, podatkovnih in
- aplikacijskih)
- namestitev, razvoj, nadzor in vzdrževanje skupnih baz podatkov
- namestitev, razvoj, nadzor in vzdrževanje diskovnih polj
- namestitev, razvoj, nadzor in vzdrževanje varnostnih kopij
- namestitev, razvoj, nadzor in razvoj lokalnih, regionalnih in mednarodnih omrežij
- testiranje novih izdelkov pred namestitvijo v okolje
- namestitev, razvoj, nadzor in vzdrževanje sistema za zagotavljanje kontrole
- dostopov (security policy)
- namestitev, razvoj, nadzor in vzdrževanje sistema za samodejno nameščanje novih
- izdelkov v okolje
- izdelava paketov za samodejno nameščanje novih izdelkov v okolje

Celotno operativno delo za namestitev novega strežnika in odjemalcev v bančno okolje je potrebno usklajevati s tem sektorjem. Vse delo se izvaja na podlagi delovnih nalogov in preko obrazcev, ki jih je predhodno potrebno potrditi pri poslovnih skrbnikih. Ti lahko zahtevajo storitve, ki jih PC Team iz tega sektorja nudi. Navadno prihaja večji del zahtevkov s strani CAU oddelka, ki ima logični pregled nad vsebino, ki je dostopna..

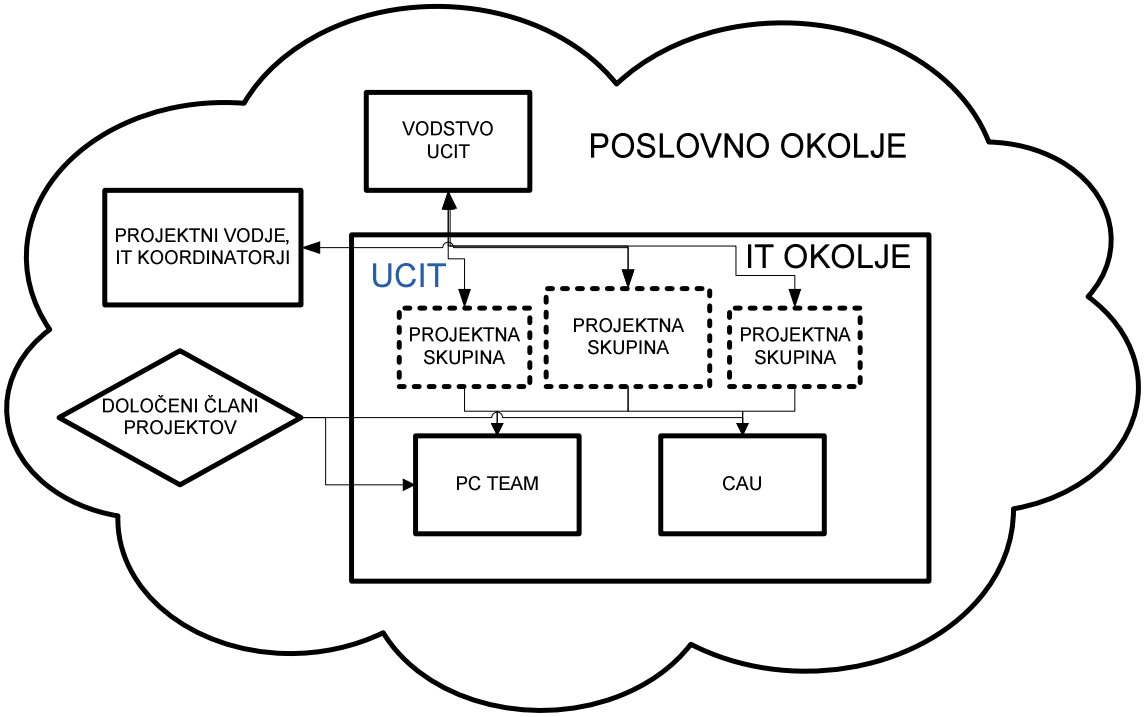
1.3.2.3 Sektor za podporne funkcije

UCIT sprejema tudi proračun za nadaljnja obdobja na podlagi skupnega proračuna, ki ga banka nameni za notranji razvoj in s tem določa, katera področja, orodja in aplikacije se bodo vpeljali in razvijali. Prav tako ima organizirano projektno pisarno za razvojne projekte in skupino trenerjev, ki so odgovorni za izobraževanje uporabnikov ob instalacijah novih

verzij programske opreme. V tem sektorju se vodi tudi finančni del za vse investicije in stroške v informacijsko tehnologijo in razvojne ter vzdrževalne projekte.

V tem sektorju je tudi skupina, ki skrbi za centralno administracijo uporabnikov – CAU (določanje pravic in mesta dostopov uporabnikom, uporabniška imena, področje delovanja, pripadnost omrežnim skupinam, določa omrežne vire, omrežnim virom določa, katere skupine jih smejo uporabljati in na kakšen način, določa elektronski poštni naslov in način uporabe elektronske pošte).

Vsak poseg glede določanja ali spreminjanja dostopov uporabnika do novih ali starih mrežnih virov je primarno naslovljen na CAU. Vsi obrazci so dostopni na intranetni strani (KIT). Obrazci se izpolnijo in glede na obrazec se zberejo različni podpisi odgovornih. Tako izpolnjen obrazec se preda CAU, ki lahko ta zahtevek prenaslovi na PC Team. Navadni uporabniki vse svoje zahteve podajo CAU, le posebej določeni IT skrbniki lahko neposredno dostopajo do PC Team-a in s tem zmanjšajo odzivni čas.



Slika 1.2: Umestitev IT okolja v poslovno okolje

1.4 Opis starega načina izdelave dokumentacije pri obeh ciljnih projektih

Kot dokumentacijo obravnavamo vse oblike zapisov. Navadno ti zapisi nastajajo v programskem paketu Microsoft Office, bolj natančno v oblikah Word in Excel. Statistično skupaj predstavljajo preko 92%, ostalih 8% pa predstavljajo Powerpoint predstavitve, čiste ASCII tekstovne datoteke in skenirani dokumenti. Datoteke ustvarjamo na novo, jih shranjujemo, si jih izmenjujemo, ali jih želimo poiskati. Po računalniški definiciji je dokumentacija organiziranana zbirka evidenc, ki opisuje strukturo, namen, delovanje,

vzdrževanje in zahtevo po podatkih za računalniški program, operacijski sistem, strojno opremo ali napravo [28]. Dokumentacija v podjetju NLB d.d. opisuje pretežno:

- tehnične specifikacije
- analize sistema
- statistike različnih zalednih sistemov (npr. Sistem za spremljanje zahtevkov)
- poročila zalednih sistemov in aplikacij (ročna in samodejna)
- zapisi o testiranju
- zapisniki sestankov
- razne razpredelnice in preglednice
- skenirani dokumenti in faksi
- korespondenca z elektronsko pošto
- navodila
- priročniki
- certifikati
- slike
- itd.

Oba projekta sta že pred mojim prihodom imela skupni omrežni vir, kjer so zaposleni shranjevali svoje dokumente. Omrežni vir je bil razdeljen približno logično in je v različnih podmapah po skupinah določal, kateri uporabniki lahko do njega dostopajo za branje ali pisanje. Različni osnutki dokumentacije so ležali v teh mapah in v imenih je bilo določeno, katero verzijo gledamo.

1.4.1 Težave s poimenovanjem datotek

Našteli bomo le najbolj tipične oblike imen datotek, s katerimi so uporabniki želeli določiti različico dokumenta. Na kratko bomo opisali slabosti takih oblik na primeru najbolj pogostega scenarija, to je iskanja zadnje različice datoteke. Primer je datoteka, ki se osnovno imenuje »datoteka.txt« in je bila spremenjena dvajsetega avgusta dvatisočega leta od uporabnika »Milica«.
Tipična poimenovanja so:

- 20-5-2008_datoteka.txt
- 08-5-20_dateka.txt
- 20080520_datoteka.txt
- datoteka-20-5-2008.txt
- datoteka_08-5-20.txt
- datoteka_20080520.txt
- datoteka-Milica.txt
- datoteka-rev0.1.txt
- datoteka_Milica_rev0.1.txt
- datoteka_20080427_rev0.1.txt

Vsi načini poimenovanja, ki se začnejo z datumom, imajo kot glavno slabost to, da je v mapi skoraj nemogoče brskati po imenu datoteke. Ureditev po imenu vedno postavi enake datume skupaj. Iskanje zadnje različice je tako zelo težko, saj je potrebno začeti na koncu

seznama datotek v mapi in se prebiti proti začetku tako, da iščemo ime datoteke »nekje na sredini« imena datoteke.

Ugotavljamo torej, da je najbolje, če označbo o reviziji postavimo na konec datoteke. V tem primeru je oblika dan-mesec-leto slaba, saj sortiranje po imenu premeša vrstni red datotek. Res, da je nabor takih datotek majhen, a obstaja možnost, da uporabnik spregleda resnično zadnjo različico. Seveda ob predpostavki, da so se vsi predhodni uporabniki, ki so delali na datoteki držali dogovora, da je datum na koncu imena datoteke. V primeru, da je različic na isti dan več, so uporabniki to reševali z dodajanjem za datumom: »-1«, »rev0.1«, lastnim imenom ipd. Taka uporaba verzioniranja ni slaba, ima pa še vedno precej pomanjkljivosti, ki jih sistemi za nadzor različic nimajo in jih elegantno rešujejo.

Nekateri uporabniki so datoteke poimenovali večinoma brez datuma, samo z revizijo na koncu, saj trdijo, da je datoteko naknadno mogoče poiskati z ureditvijo datotek v raziskovalcu po datumu. Argument proti temu je primer, ko nek uporabnik ponovno shrani datoteko in ji s tem popravi atribut datuma v datotečnem sistemu.

Vsi zgoraj našteti načini shranjevanja revizij imajo vsekakor to slabost, da mapa hitro vsebuje veliko število datotek. Tako je iskanje natanko enega stanja datoteke zelo oteženo in zahteva, da uporabnik pregleda praktično vse datoteke v mapi, preden izve, katera je zadnja različica, na kateri lahko dela dalje, ali pa jo mora spreminjati. Delna rešitev je uvedba mape arhiv kot pod-mape glavne mape. Tam se premikajo vse datoteke, ki so starejše od te, na kateri trenutno delamo. Oba projekta sta v mapah, ki so bile najbolj obsežne po številu datotek, izbrala to rešitev. Žal ponovno brez enotne izbire imena takega arhiva, kar bi naknadno otežilo avtomatizacijo prenosa na nov sistem verzioniranja.

1.4.2 Težave z dostopanjem in spreminjanjem istega dokumenta s strani več različnih uporabnikov

Pri projektih, na katerih je sprva delalo le nekaj točno določenih uporabnikov se rado zgodi, da uporabniki ne želijo oddati od sebe dokumenta, ki ni zadnja različica. Take dokumente shranjujejo lokalno pri sebi. V primeru, da jo pošljejo v revizijo kateremu od drugih članov projekta in želijo dodatke ali popravke, imajo odgovore navadno shranjene kar v poštnem predalu elektronske pošte. Člani projekta nimajo jasno določenih strategij za razvrščanje pošte po mapah, ki logično razmejujejo projekte in zadeve. Velikokrat se dogodi, da je neka elektronska pošta preprosto spregledana ali izbrisana preden so spremembe vključene v dokument, ki ga član upravlja. Poleg tega je možnost izgube takih dokumentov velika, saj ni celotnih varnostnih kopij. V primeru bolezni ali dopusta pa je sodelavcem izjemno težko, če ne nemogoče, nadaljevati delo. Čeprav so na razpolago skupni omrežni viri, jih večina zaposlenih ne želi uporabljati, saj se ravno tam bojijo izgube podatkov in tega, da datoteke ne bodo več našli.

Poimenovanje brez jasne strategije je povzročalo dodatno delo tistim uporabnikom, ki so morali datoteko poiskati. Obenem so se pojavile napake, ker uporabnik ni našel prave datoteke in je tako preprosto izgubil vse, kar je bilo medtem narejenega na datoteki. Enotna strategija, ki je jasna vsem uporabnikom, ki na dokumentaciji delajo ali jo uporabljajo, je tako ključnega pomena za odpravo nepotrebnih napak.

1.4.3 Težava pri nadzoru dela na projektu

Zaradi prejšnje težave imajo projektni vodje in vodje skupin zelo otežen pregled nad tem, kako in na katerih področjih delo napreduje. Različni zaposleni imajo različno dinamiko shranjevanja in odlaganja na skupne omrežne vire, kar povzroča ogromno dela vsem, ki poskušajo nadzorovati in spremljati tako delo. Nekateri zaposleni šele po več tednih dajo svoje dokumente na voljo tudi ostalim članom . Za vodjo projekta je, če mu ni omogočeno sprotno spremljanje in normiranje opravljenega dela zaposlenih, koliko časa bo določen korak v projektu trajal, ocena izjemno otežena . Zahtevati od zaposlenih, da uporabljajo to strategijo discipliniranega odlaganja svojih dokumentov je težavno s psihološkega vidika. Vodja projekta prav tako porabi veliko časa, da sproti preverja, ali so zaposleni datoteke spreminjali.

1.4.4 Težava počasnosti obnovitve izgubljenih ali izbrisanih podatkov

Operacijski sistem Windows je nekoliko neroden, saj z izjemno lahkoto omogoča, da uporabnik z miško pomotoma odnese celo mapo datotek na neko drugo mesto. Ravno tako uporabniki večkrat pomotoma brišejo svoje stare datoteke, saj ne vedo, da jih tudi drugi še potrebujejo, ali da jih bodo potrebovali tudi sami. V takem primeru obstaja točno določen predpis, ki določa na kakšen način uporabniki zaprosijo za obnovitev (ang. restore) takih datotek ali map. Uporabnik na intranetnih straneh CAU najde obrazec za obnovitev in ga izpolni po naslednjem postopku:

- kdo zahteva
- kje je stara lokacija dokumenta ali mape (omrežni vir ali uporabniška mapa na lokalni delov
- kdaj je okvirni čas izbrisa
- pridobi podpise vseh odgovornih nadrejenih (linijski vodja, projektni vodja, vodja skupine).

Pravilno izpolnjen obrazec se osebno odda CAU, ki ga dodeli tistemu administratorju znotraj PcTeam-a, ki je odgovoren za varnostne kopije strežnika, s katerega je datoteka ali mapa izginila. Ta ga glede na čas izginotja lahko išče med dnevnimi, tedenskimi ali mesečnimi varnostnimi kopijami. Vsaka od teh varnostnih kopij je lahko na različni obliki medija (diskovno polje, CD, trak,...). Kratkoročne varnostne kopije so kumulativne in že zaradi tega je obnova zelo počasna. Povprečen čas od oddaje uporabnikove vloge za obnovo in dejansko obnovo je navadno 2 do 3 dni. Če datoteka nima prave vsebine, se celoten postopek ponovi. Uporabniki obnovo zaradi tega dolgotrajnega postopka uporabljajo le kot skrajno rešitev in še to neradi.

1.4.5 Težava varovanja podatkov in določanja dostopov

Banka je velik sistem, v katerem v različnih vlogah deluje veliko ljudi. Vsi podatki niso namenjeni vsem zaposlenim. Nekateri morajo podatke le brati, drugi jih lahko tudi spreminjajo, tretji pa o njihovem obstoju sploh ne smejo vedeti. Določanje uporabniških dostopov do omrežnih virov poteka preko posebnega obrazca CAU, ki vsebuje podatke:

- kdo zahteva spremembo dostopa
- zakaj zahteva spremembo dostopa

- na katero uporabniško skupino in omrežni vir je vezan dostop
- katere so zahtevane pravice dostopa
- pridobi podpise vseh odgovornih nadrejenih (linijski vodja, projektni vodja, vodja skupine).

V primeru, da je omrežni vir že vzpostavljen in uporabniška skupina že ustvarjena, izpolnjevanje zahtevka traja en dan. V nasprotnem primeru je zahtevkov več in čas sorazmerno daljši. Sama tehnična izvedba je preprosta, vendar trajanje dogovora o dostopnih pravicah z odgovornimi dolgotrajno, saj je vsaka sprememba dobro preučena.

1.4.6 Težave s pregledovanjem razlik med dvema različicama istega dokumenta

Pri vključevanju različnih popravkov v osnovni dokument je najprej potrebno ugotoviti, kaj je v prejeti različici drugačnega v primerjavi z osnovnim dokumentom. Microsoft Word vsebuje nekaj sistemov za pregledovanje različic. Dva najbolj poznana sta funkcionalnosti »Track Changes« in »Diff Two Documents«.

Funkcionalnost »spremljaj razlike« je potrebno omogočiti takoj na začetku ustvarjanja dokumenta in nihče od naslednjih uporabnikov, ki delajo na tem dokumentu, je ne sme izklopiti. Funkcionalnost je dovolj dobra, dokler si dokument podajata dva uporabnika. V primeru, ko eden izmed uporabnikov prejme popravke iz več virov naenkrat, pa le-ta v praksi odpove. V tem primeru si uporabniki navadno dokumente v celoti natisnejo in pretipkajo popravke v osnovni dokument. Enako storijo tudi v primeru, ko te funkcionalnosti ne poznajo.

Funkcionalnost »razlikuj med dvema dokumentoma« je zelo redko poznana uporabnikom. Omogoča, da iz osnovnega dokumenta izberemo še enega in ta nam (oblikovno podobno funkcionalnosti »spremljaj razlike«) pokaže razlike. Funkcionalnost je primerna za uporabo, če »spremljaj razlike« ni omogočena na dokumentu.

Kljub obema možnostma za Wordove datoteke je to zelo pereč problem pri uporabnikih, poleg tega pa v okviru programskega paketa Microsoft Office ni rešitev za datoteke Excel in prezentacije Powerpoint. Enako velja za ostale tipično uporabljane oblike datotek.

1.4.7 Težava z določanjem natančnega časa nastanka datoteke

Velikokrat želimo vedeti, kdaj natančno je nek dokument nastal. Težave s potvorjenim datumom so na skupnih omrežnih virih precej pogost pojav. Do spremembe datuma pride velikokrat zaradi tega, ker nekdo ob prebiranju dokumenta popravi kakšno manjšo slovnično ali vsebinsko napako. Ko tako popravljen dokument išče nekdo drugi, mu novi datum pomeni težavo pri ugotavljanju dejanskega časa nastanka dokumenta.

Datoteke Excel so glede spremembe datuma še posebej težavne. Ob vsakem odpiranju datoteke Excel aplikacija spremeni datum »Date modified« v datotečnem sistemu na trenutni čas. Če datoteke nismo shranili, aplikacija vrne datum in čas na prejšnji zapis, v nasprotnem primeru ostane čas spremembe, čas zadnjega shranjevanja. Težava nastane, če

je kateri od uporabnikov na skupnem omrežnem viru imel odprto datoteko in se mu poruši računalnik, ali se aplikacija sesuje. V tem primeru (poleg tega, da zelo pogosto datoteka ostane celo zaklenjena za pisanje s strani drugih uporabnikov) ostane v datotečnem sistemu zabeležen čas zadnje spremembe - trenutek odpiranja datoteke.

Najpomembnejše datoteke ravno zaradi tega vsebujejo glave, ki vsebujejo datum spremembe, ki se mora ročno posodobiti na dejanski čas. Ravno tako imajo (žal le zares pomembni dokumenti) na začetku tabelo, v kateri je vpisan čas spremembe, avtor in namen spremembe.

1.4.8 Težava z določanjem tega, kdo je opravil popravek v datoteki

Vsebine datotek so občutljivo področje. Ne želimo, da jih lahko kdorkoli spreminja, ne da bi vedeli, kdo je to bil. V navadnem datotečnem sistemu lahko vemo le, kdo je bil zadnji, ki je dokument popravljal. Lahko se nam zgodi, da nekdo izbriše ali priredi cele odlomke besedil. Če bo za njim še kdorkoli to datoteko ponovno shranil, nikoli ne bomo vedeli, kdo je bil tisti, ki je nek odlomek spremenil ali izbrisal. Funkcionalnost »Spremljaj spremembe« je na voljo le za datoteke Word. Žal z njo ne moremo pokriti vseh varnostnih vidikov, ki so potrebni za resen nadzor nad avtorstvom sprememb, saj je funkcionalnost namenjena dobronamerni uporabi. Zlonameren uporabnik jo lahko mimogrede zaobide, ali še huje potvori.

1.4.9 Težava z določanjem vzroka za nastanek nove revizije

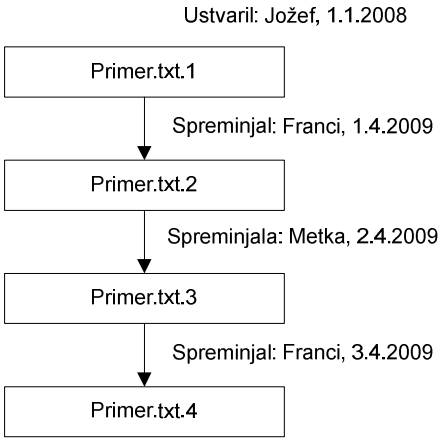
Najpomembnejše datoteke imajo v svojem začetku tabele, ki služijo kot dnevnik zapisov. Vsebujejo edinstveno oznako revizije, datum, avtor in namen spremembe. Pomembno je, da obstaja jasen dogovor, da uporabniki beležijo svoje spremembe v take tabele. Velikokrat so spremembe manjše in jih zato uporabniki ne želijo vpisati v to tabelo. Opis namena je seveda zelo pomembno polje, žal pa je tudi edino polje, ki se le redkokdaj izpolni. Če je že izpolnjeno, je opis zelo skop ali »kriptičen« (razumljiv samo avtorju).

2 IZPOPOLNJEN SISTEM HRANJENJA DOKUMENTACIJE

Po temeljiti opredelitvi težav starega sistema shranjevanja dokumentov in ostalih vrst datotek na skupne omrežne vire, je naravna rešitev večjega dela težav uvedba sistema za nadzor različic na nivoju celotnih projektov. Najprej je potrebno opredeliti kriterije, po katerih bomo izbirali orodja, nato bomo med vsemi orodji izbrali tista, ki bodo kriterijem izbire najbolje ustrezala, na koncu pa bomo izbrali še metodologijo in opredelili jasna pravila, kako sistem uvesti na projekte in ga približati uporabnikom.

Sistem za nadzor različic je v IT okolju dobro poznano orodje, ki služi za shranjevanje zgodovine razvoja. Najdlje se uporablja v programskem inženirstvu, najpogosteje za shranjevanje izvorne kode in konfiguracijskih datotek. Razvijalec mora imeti možnost potegniti iz arhiva staro različico in jo popravljati. Težave, ki se pojavljajo pri shranjevanju programske kode in konfiguracijskih datotek, so zelo podobne tem, ki jih srečamo pri shranjevanju dokumentacije.

Sistem za nadzor različic je upravljanje večjega števila revizij iste enote informacije. Najpogosteje se uporablja pri inženiringu in razvoju programske opreme za vodenje, razvoj na digitalnih dokumentih, kot je izvorna koda aplikacij, umetnostnih virih, kot so načrti ali elektronski modeli in za ostale projekte, pri katerih istočasno dela skupina ljudi. Spremembe teh dokumentov so navadno identificirane z inkrementiranjem nekega števila ali kode znakov. To imenujemo številka revizije, stopnja revizije ali najpreprosteje revizija. Zgodovinsko jo povežemo z osebo, ki je naredila spremembo. Najosnovnejši način spremljanja revizij je, da pri nastanku nekega dokumenta temu damo revizijsko številko 1. Ko je narejena prva sprememba, se revizija poveča na dva in tako dalje (slika 2.1). [7]



Slika 2.1: Preprost način verzioniranja dokumenta Primer.txt

Sistem za nadzor različic je najpogosteje samostojna aplikacija, vendar je nadzor revizij tudi že vključen v različne tipe programske opreme kot so urejevalniki besedil (npr. Microsoft Word, OpenOffice.org Writer, Koffice, Pages, Google Docs), preglednice (npr.

OpenOffice.org Calc, Google Spreadsheets, Microsoft Excel) in različni CMS. Integrirana kontrola revizije je ključna funkcionalnost pri vseh teh skupinah, saj najpogosteje omogoča vračanje vsebine na prejšnje stanje, kar je ključno za urejevalce, da lahko spremljajo spremembe drug-drugega, popravljajo napake in preprečujejo namerne vandalizme zlonamernih popravljavcev. [7]

Različni avtorji, ki analizirajo razvojne branže, vedno bolj priznavajo pomembnost in nujnost sistemov nadzora različic pri organizaciji projektov z več razvijalci [18].

2.1 Določanje kriterijev izbire

Če želimo objektivno izbrati, moramo pri vsakem odločanju, določiti jasne kriterije izbire in jim določiti primerno pomembnost. Glede na že opisano okolje in težave sem izbral tiste kriterije, za katere sem ocenil, da bodo doprinesli koristen prispevek k napredovanju splošnega dela na projektih. Našteti so vsi kriteriji, ki lahko vplivajo na odločitev, ni pa nujno, da bo potrebno vse upoštevati.

Kriteriji izbire orodja:
a) Model skladišča
Poznamo tri različne oblike skladišč, ki jih taki sistemi premorejo.

- lokalno skladišče
- distribuirano skladišče
- sistem odjemalec-strežnik

Sistem odjemalec-strežnik je sistem, pri katerem uporabniki, razvijalci, delijo skupno centralno skladišče. Lokalno skladišče pomeni, da je skladišče postavljeno na lokalni datotečni sistem. Priklop z drugih delovnih postaj ni možen. Distribuirano skladišče je sistem, pri katerem navadno uporabniki delajo na svojem lokalnem skladišču in v posebnem koraku svoje delo in spremembe delijo z ostalimi lokalnimi skladišči.
Prednost bomo dali sistemu odjemalec-strežnik, saj bi kateri od drugih sistemov zahteval več vzdrževanja, večjo discipliniranost uporabnikov, bistveno večjo potrošnjo prostora in predvsem večjo težavnost namestitve. Poleg tega sta preostala sistema bolj primerna za nepovezane uporabnike, ki si občasno izmenjujejo informacije. Ta kriterij je pri naši odločitvi zelo pomemben.

b) Vključenost v okolje
Bodoči uporabniki ne smejo čutiti ostrega prehoda med starim in novim sistemom. Novi sistem mora biti na videz dobro integriran v ostale že obstoječe sisteme, ki so jih uporabniki že vajeni.

c) Preprostost uporabe tudi za najbolj neuke uporabnike
Za vsako novo orodje, sistem ali aplikacijo je idealno, da je njegovo delovanje mogoče razložiti v nekaj preprostih stavkih z razlago nove terminologije vred. Orodje, ki je uporabljeno za izvajanje novega sistema mora biti intuitivno. Onemogočati mora nepravilno delovanje in uporabnika opozoriti pred nezaželenimi ali potencialno nevarnimi akcijami. Pod ta kriterij bom umestil tudi podporo poslovenjenju uporabniškega vmesnika, saj bo s tem orodje lažje dostopno vsem vrstam uporabnikov in bo prehod nanj preprostejši.

d) Stabilnost aplikacije
Želimo si, da nobena aplikacija ne bi delovala slabo, se rušila, povzročala memory leak-ov, vplivala na druge aplikacije in zaledne sisteme ali ogrožala delovanje operacijskega sistema, na katerem deluje. Pod ta kriterij bom uvrstil tudi promet, ki ga na omrežju povzroča aplikacija/orodje pri velikem številu uporabnikov in velikem številu aplikacij, ki jih le-ti uporabljajo. Vsak dodaten promet predstavlja infrastrukturno tveganje.

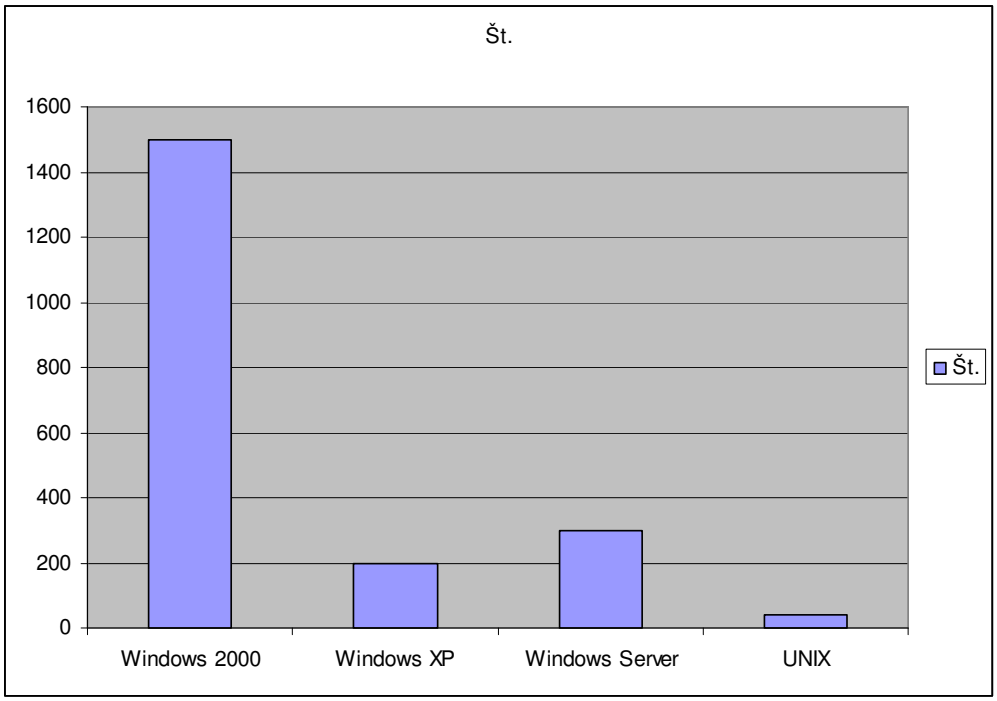
e) Cena
Cena novega sistema je večplastno sestavljen kriterij. Na ceno vpliva cena morebitnih licenc samega orodja, njegovih dodatkov in opreme, s pomočjo katere teče. To pa je le prvi najbolj očiten strošek. Poleg tega kot strošek nastopa tudi število inženirskih ur, ki so potrebne za namestitev novega sistema, kar pomeni, kako hitro je mogoče sistem namestiti in nastaviti. Nenazadnje je v ceno potrebno uvrstiti tudi strošek truda, časa in delavnic, ki so potrebne za usposobljenost uporabnikov za delo z novim sistemom. Velja nenapisano pravilo: »Dražja je licenca, hitrejša je namestitev in cenejši je strokovnjak - cenejša je licenca, počasnejša je namestitev in dražji je strokovnjak«. Če je sistem sestavljen iz odprtokodnih rešitev, to navadno pomeni, da bomo toliko več denarja in časa morali investirati v samo namestitev, nastavljanje in ustvarjanje uporabniške dokumentacije.

f) Licenciranje in pravice uporabe orodij
V okvir cene bi lahko uvrstili tudi licenciranje. Ker pa ne vemo, koliko uporabnikov bo orodje v resnici uporabljalo, je nedvoumno bolj prav, če opredelimo ta kriterij kot posebno kategorijo in jo posebej uravnotežimo. V primeru, da bo uporabnikov orodja malo, je cena irelevantna napram razliki v času, ki je potrebnen za namestitev komercialne celostne rešitve (all-in-one). V nasprotnem primeru se podaljšanje časa, ki ga potrebujemo za namestitev odprtokodnega orodja pod licenco GNU/GPL, ali katero od različic izplača pri večjemu številu uporabnikov.

g) Varnost
Bančno okolje že po pravilu potrebuje višjo stopnjo varnosti omrežja, aplikacij in operacijskega sistema. Gre za denar in osebne podatke komitentov. Poleg tega izpad ali vdor v katerega od sistemov lahko predstavlja najmanj naslednje posledice: izgubo licence Banke Slovenije, visoke denarne kazni države, visoke denarne tožbe komitentov in ali drugih fizičnih ali pravnih oseb, izgubo komitentov ali izgubo zaupanja komitentov [24]. Izbrani sistem bo, preden bo prejel uspešen »Zapis o testiranju« s strani UCIT , seveda naprej dobro testiran s strani banke. Varnost mora biti zagotovljena z neprekinjenim preverjanjem identitete uporabnika na domenskem krmilniku. Kanali, preko katerih se prenašajo datoteke in identifikacija uporabnika, morajo biti varni.

h) Operacijski sistem (heterogena okolja)
Operacijski sistemi znotraj podjetja NLB d.d. so pretežno homogeni na enaki ravni. To pomeni, da so delovne postaje pretežno poenotene glede operacijskega sistema in nameščenih aplikacij. Večina uporabnikov bo torej uporabljala orodje nameščeno na operacijskem sistemu Windows 2000. Nekaj uporabnikov dela s pomočjo operacijskega sistema Windows XP Professional in nekaj administratorjev potrebuje ta sistem pri operacijskem sistemu Windows 2003 Server. Upoštevati moramo tudi najboljši možen scenarij in predvideti uporabo tudi na ostalih strežnikih, kjer bi lahko administratorji potrebovali sistem za nadzor različic za verzioniranje konfiguracijskih datotek, brali navodila ali odlagali poročila. Vse slednje se lahko izvaja ročno ali avtomatizira preko

skript. Velikostni red teh namestitev je preko 1500 Windows 2000, 200 Windows XP, 300 Windows Serverjev, in 40 različnih operacijskih sistemov iz družine UNIX (AIX, DEC ...) [15] (slika 2.1).



Slika 2.2: Število različnih operacijskih sistemov v bančnem okolju

Iz zgoraj navedenega je razvidno, da iščemo programsko rešitev, ki bo nudila najboljšo podporo za odjemalce na sistemu Windows. Za strežnik je tudi najbolje, da deluje na operacijskem sistemu Windows. V NLB d.d. je največ administratorjev z dobrim poznavanje tega strežnika, pa tudi strežnikov, ki že delujejo in na katerih bi bilo mogoče namestiti strežniško aplikacijo, je veliko. Vendar izbira operacijskega sistema za strežnik ni nujno omejena na Windows Server, torej je možna končna izbira tudi za kateri drugi operacijski sistem.

i) Masovno nameščanje, nastavljanje in nadgrajevanje

Že iz zgornjega kriterija je razvidno, da je število računalnikov, na katere je potrebno namestiti orodje za nadzor različic, veliko. Glede na to, da je to samo eno izmed orodij, ki so potrebna za delo uporabnikov, je nerealno pričakovati, da bi zaposleni PCTeam-a porabili nekaj tednov za namestitev tega orodja na različnih geografskih lokacijah. Gre torej za nujno zahtevo, brez katere ne moremo pričakovati, da bo UCIT karkoli sprejeto. Bančni sistem je že pripravljen za tako množično nameščanje, nastavljanje in nadgrajevanje.

Za operacijske sisteme Microsoft obstaja rešitev v obliki SMS – System Management Server proizvajalca Microsoft. Izdelek upravlja velike skupine računalnikov z nameščenim operacijskim sistemom Windows. Upravljanje s konfiguracijo nudi oddaljen nadzor, nameščanje popravkov, distribucijo programske opreme, nameščanje operacijskega sistema in samodejno ustvarja seznam strojne in programske opreme na nivoju celotnega podjetja. [4]. V banki se s podmnožico zgoraj naštetih funkcionalnosti SMS dopolnjuje z aplikacijo

ScriptLogic proizvajalca QuestSoftware. Ta je uporabljan za administracijo omrežij na osnovi operacijskih sistemov Microsoft Windows. Dovoljuje administratorjem omrežja oddaljeno upravljanje na mrežo priklopljenih osebnih računalnikov in strežnikov in s tem zagotavlja uniformnost omrežnih konfiguracij in uveljavljanje varnostne politike[5]. Seveda pa je lasten sistem nameščanja ravno tako dober kot že izbrana rešitev, vendar mora pokrivati vse ključne funkcionalnosti že obstoječega bančnega sistema. Osredotočili se bomo na oceno avtomatizirane namestitve v operacijskih sistemih Windows, ker je število ostalih dovolj majhno, da je tamkajšnja ročna namestitev dovolj hitra (priprava avtomatizacije traja dlje kot namestitev na vse potrebne sisteme). Dodatno lahko h končni oceni tega kriterija doprinese morebitna možnost samodejne integracije v Microsoft Office, IDE, Windows Explorer ipd.

j) Ciljni uporabniki
So iz obeh skupin bančnih uporabnikov. To so poslovni in razvojni uporabniki. Prvi bodo nov sistem uporabljali za verzioniranje, ustvarjanje in pregledovanje dokumentacije. Njihovo poznavanje informacijskih tehnologij je omejeno. Ne moremo pričakovati velikih prilagoditev glede metodologij dela. Razvojni uporabniki bodo dokumentacijo ustvarjali in pregledovali ter dodatno uporabljali za verzioniranje konfiguracij. Projektni vodje pa potrebujejo poglobljeno znanje sistema in bodo funkcionalnosti primerne njihovemu delu in potrebam ocenjevali z drugim kriterijem (možnost poročanja).

k) Administracija
Mora biti lahka in transparentna. Krivulja učenja mora biti čimbolj strma in kratka. Za najzahtevnejše posege in potrebe banka navadno uporablja zunanje svetovalce, s katerimi si pomaga in s tem ceni administracijo, saj njihovim administratorjem poglobljeno poznavanje sistema ni potrebno .

l) Razreševanje težav prejšnjega sistema shranjevanja in verzioniranja
Oceniti moramo, v kolikšni meri novi sistem verzioniranja odpravlja težave starega sistema. Pod ta kriterij za skupno oceno pri izbiri orodja bomo uvrstil tudi parcialno možnost avtomatizacije začetnega stanja skladišča, s katerim bomo poskusili odpraviti pomanjkljivost starega sistema urejanja dokumentacije. Obenem bomo na ta način uporabnike poskusili naučiti novega sistema, saj njihove datoteke ne bodo več poimenovane enako kot prej in ne bodo na istih lokacijah.

m) Obveščanje
Za nekatere datoteke je dobro, da je določen krog uporabnikov, če pride do sprememb, obveščen v naprej . Navadno so to datoteke s podatki o tem, kdaj bo zaposleni na letnem dopustu, zbirna poročila, pravila in navodila pri razvoju ipd.. V vsakem primeru pa je potrebno v okviru skupin uporabnikov o spremembah obvestiti vodjo, saj na ta način lahko uspešno spremlja in normira delo.

n) Statistike in poročila
Orodje v idealnih okoliščinah nudi način za poročanje vodstvu projekta na najvišjem nivoju. Če je dokumentacija pravilno razvrščena po področjih poslovanja, izdelkih, modulih izdelkov, skupinah uporabnikov (vlogah) in logični fazi razvoja, je s statistikami in poročili mogoče na najvišjem nivoju dobro videti napredek na različnih projektih. S tem kriterijem poskušamo oceniti, ali ima orodje že nek samodejni način in ali je potrebno morda

osnovnemu orodju dodati že izdelano rešitev ali morda do-programirati aplikacijo oziroma sistem, da je sposoben takih poročil.

o) Pregledovanje razlik največjega možnega števila oblik datotek
Idealno je, da sistem že zna delati z največjim možnim številom oblik datotek. Za oblike datotek, ki niso podprte, mora obstajati preprost mehanizem, s katerim lahko dodajamo programe, s katerimi bo možno te razlike pregledovati. Najbolj potrebujemo sistem, ki bo imel podprte oblike datotek Microsoft Office, zato bo sistem, ki bo imel boljšo podporo za te oblike najbolje ocenjen.

p) Kader, ki dobro pozna in ima izkušnje z integracijo določenega orodja
Pri vsaki integraciji in nasploh vsakem delu, ki se ga lotevamo smo vedno omejeni s kadri, ki jih imamo na voljo za izvedbo. Pomembno je, da imamo za izbrani sistem na voljo kader, ki sistem pozna in ima z njim izkušnje. Če kadra nimamo, moramo vsaj vedeti, kje tak kader lahko pridobimo za trajno, občasno ali enkratno sodelovanje.

r) Rešene ostale tipične težave in interni mehanizmi sistema za nadzor različic
Sistemi za nadzor različic so zrela skupina izdelkov. Zaradi tega obstaja precej kriterijev, ki uporabniku niso najbolj očitni. Ni pomembno, da ima nek izdelek realizirane vse funkcionalnosti, da bi bil boljši. Gre bolj za uravnoteženost funkcionalnosti in seveda možnost, da pomanjkanje le-teh ne zavira nemotenega dela.
Na grobo delimo te kriterije [6]:

- Operacije nad skladiščem
- Tehnična dovršenost
- Uporabniški vmesniki

<u>Operacije nad skladiščem, ki so pomembne so:</u>

Nedeljivo objavljanje
Podpora nedeljivemu objavljanju pomeni, da skladišče ne bo ostalo v nekonsistentnem stanju, če je bila operacija nad skladiščem prekinjena med izvajanjem,.

Preimenovanje in premikanje datotek in map
Sistem mora premikanje ali preimenovanje mape ali datoteke beležiti v zgodovini in ohraniti staro zgodovino mape.

Pametno zlivanje po preimenovanju in premikanju
Sistem mora spremljati in beležiti preimenovanja in zlivati zgodovine in vsebine datotek.

Zaznavanje kopij datotek in map
Ali zna sistem hraniti zgodovino na enem mestu v primeru, da je ena datoteka na več mestih v skladišču?

Podvojitev oddaljenega skladišča
Ali sistem podpira kloniranje oddaljenega skladišča tako, da pridobi funkcionalno ekvivalentno kopijo na lokalnem sistemu in ali je to možno storiti brez posebnih pravic dostopa do oddaljenega skladišča.

Prenašanje sprememb na krovno oddaljeno skladišče
Ali sistem omogoča širjenje sprememb iz enega skladišča na drugega?

Pravice nad skladiščem
Ali je mogoče določiti pravice dostopa do dela skladišča in ali je dostop odprt za vse?

Podpora »množice sprememb«
Ali skladišče podpira večje število hkratnih sprememb? [6,3]

Spremljanje zgodovine na nivoju datoteke
Ali ima sistem za nadzor različic možnost spremljanja zgodovine na nivoju datoteke »vrstico-po-vrstico«? Ali lahko na primer za vsako vrstico v datoteki na preprost način pokaže, v kateri reviziji je bila nazadnje spreminjana in od koga?

Praktični dodatki
Možnost dela na le eni mapi v skladišču
Ali lahko sistem prevzame le eno mapo skladišča in omeji objave le na eno mapo?

Spremljanje neobjavljenih sprememb
Ali lahko programska oprema pregleduje razlike, ki še niso objavljene v skladišču?

Opomba ob objavi na nivoju datoteke
Ali sistem omogoča zapis opombe v zgodovino datoteke ob objavi *množice sprememb* in dodatno še k zgodovini *množice sprememb*.

Tehnična dovršenost
Dokumentacija
Kako dobro je sistem dokumentiran in ali ga je lahko začeti uporabljati?

Preprostost namestitve
Kako preprosta je namestitev, kakšni so predpogoji in soodvisnosti in kako težko jih je urediti?

Nabor ukazov
Kako obširen je nabor ukazov in kako je kompatibilen z naborom ukazov CVS (defakto odprtokodni standard)?

Podpora omrežjem
Kako dobra in s kolikšno stopnjo je možna integracija v obstoječe sisteme omrežij (protokoli in infrastrukturo)?

Prenosljivost
Kako prenosljiv je sistem na druge operacijske sisteme, računalniške arhitekture in druge tipe sistemov?

Uporabniški vmesniki
Vmesniki za brskalnik
Ali so na voljo vmesniki primerni za brskalnike, s katerimi je mogoče pregledovati drevo revizij, zgodovino, izvajati preglede razlik ipd.?

Koliko grafičnih vmesnikov je na voljo
Koliko različnih grafičnih vmesnikov je na voljo za izbirani sistem za nadzor različic?

2.2 Ocena različnih orodij

Ponudba različnih orodij je velika, saj so sistemi za nadzor različic nastali v 70-ih letih prejšnjega stoletja. Prvi tak sistem je bil SCCS, nasledil ga je sistem RCS, tega pa CVS. Razvijalci CVS so kasneje želeli popraviti nekatere največje pomanjkljivosti tega sistema in so ustvarili CVSNT. V zadnjih dveh letih poskušajo odpraviti tudi nekatere pomanjkljivosti CVSNT in ponovno se je del razvijalcev odločil, da bo ustvaril nov izdelek EVS, ki pa še ni na voljo niti v alfa različici. To je nekako glavna veja razvoja te skupine sistemov. Na vsaki točki razvoja pa je peščica razvijalcev začela razvijati svoje orodje za svoje potrebe. Ti so navadno odpravljali najbolj pereče probleme, ki so se tisti trenutek pojavljali zaradi nezrelosti teh sistemov.

Zaradi take množice orodij bomo s pomočjo v prejšnjem poglavju opisanih kriterijev poskušali izbiro orodja postopoma zmanjševati. S tem si bomo prihranili raziskovanje vsakega od orodij, ki že v najbolj pomembnih kriterijih ne dosegajo standardov, ki jih zahtevamo.

Po temeljiti raziskavi te skupine programske opreme je na voljo 49 programskih rešitev, ki nam omogočajo kvaliteten nadzor različic:

- AccuRev
- Aegis
- Aldon
- Alienbrain
- AllChange
- AllFusion Harvest Change Manager
- Arch
- AVS
- Bazaar
- BitKeeper
- ClearCase
- CM+
- CMSynergy
- Code Co-op
- Co-Op
- CVS
- CVSNT
- Darcs
- DesignSync
- Git
- GNU arch
- LibreSource Synchronizer
- Mercurial
- Monotone
- OpenCM
- Perforce

- Plastic SCM
- PureCM
- Razor
- SourceAnywhere
- SourceHaven
- StarTeam
- Subversion
- Superversion
- Surround SCM
- svk
- Synergy
- Team Foundation Server
- Vault
- Vesta
- Visual SourceSafe

2.2.1 Kriteriji za izbiro orodja

Da bi si olajšali možnost filtriranja in hitreje prišli do ožjega izbora možnih programskih rešitev za naše zahteve, smo ustvarili tabelo [priloga 1]. Tabela vsebuje zbrane podatke, ki smo jih dobili pri referenčnih primerjalnih ocenah na straneh Wikipedije[3] in better scm-ja[6]. Ne vsebuje vseh kriterijev, ki smo jih določili v prejšnjem poglavju, vendar bomo videli, da je ob primerni izbiri, glede na to, s pomočjo katerih kriterijev želimo to razpredelnico filtrirati, možno najti optimalno rešitev zelo hitro in brez celotnega poznavanja vseh ponujenih programskih rešitev. Ti kriteriji so: vzdrževalec, stopnja zrelosti, model skladišča, model konkurenčnosti, licenca, podprte platforme, cena, programski jezik, model zgodovine, identifikacija revizij, velikost skladišča, podprti mrežni protokoli, nedeljiva objava, preimenovanje datotek, zlivanje preimenovanja datotek, simbolični linki, možnost proženja zank ob dogodkih, označene revizije, sledenje zlivanja, konverzija EOL, oznake, večjezičnost, zgodovina, pomembni uporabniki, obstoj vmesnikov za brskalnike, obstoj samostojnih uporabniških vmesnikov in integracija ali vtičniki za IDE.

2.2.1.1 Pomanjkanje dostopnih podatkov

V prvem koraku bomo takoj izločili tiste programske rešitve, o katerih ne moremo zbrati dovolj podatkov. Pri izbiri programskih rešitev je že to dovolj dober pokazatelj kakovosti in zrelosti izdelka. Če orodja večina avtorjev sploh ne obravnava pri vrednostih primerjavah in ga ne navaja v testih, lahko sklepamo, da je razlog za to slaba kakovost ali nezrelost izdelka. Želimo torej priznan izdelek, ki je dobro podprt in o katerem bomo našli dovolj uporabniške dokumentacije. Za tako programsko rešitev z večjo verjetnostjo obstaja obširna skupnost uporabnikov ali plačljiva strokovna tehnična podpora, ki nam lahko pomaga v primeru zastojev ali težav. Iz seznama v prilogi 1 bomo izločili vse tiste programske rešitve, za katere nismo mogli najti odgovorov na več kot polovico opazovanih kriterijev in podatkov, ki smo jih zbrali.

Na podlagi tega kriterija smo izločili orodja: Aegis, Arch, AVS, BitKeeper, CM+, CMSynergy, Codeville, Co-Op, Endevor, FileHamster, MKS, Mogware, OpenCM, Revision Control System, Source Code Control System, Superversion, Synergy in Vesta.

Ustrezni kandidati

Število podatkov / Ime
27
AllFusion Harvest Change Manager
Bazaar
ClearCase
DesignSync
Git
Perforce
Subversion (SVN)
Surround SCM
Team Foundation Server
26
AccuRev
AllChange
CVS
CVSNT
Mercurial
PureCM
Razor
StarTeam
25
Code Co-op
LibreSource Synchronizer
Monotone
PlasticSCM
Vault
Visual SourceSafe
24
darcs
SourceAnywhere Hosted
23
GNU arch
22
Aldon
21
Alienbrain
SVK
19
Telelogic Synergy
18
SourceHaven

Izločeni kandidati

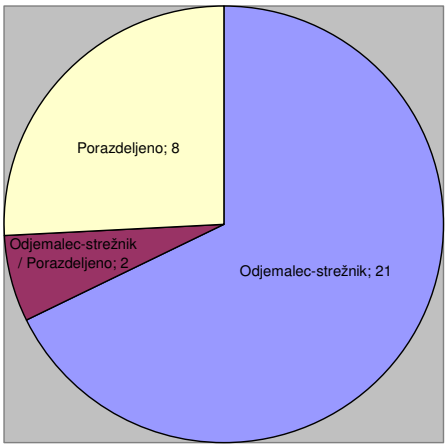
Število podatkov / Ime
12
AVS
9
BitKeeper
7
Codeville
FileHamster
MKS
Mogware
6
Endevor
Synergy
3
Arch
Revision Control System
Source Code Control System
1
Aegis
0
CM+
CMSynergy
Co-Op
OpenCM
Superversion
Vesta

Tabela 2.1: Seznam vseh programskih rešitev združenih po številu podatkov, ki smo jih zbrali

Po pregledu izločenih programskih rešitev je razvidno, da so izločene rešitve večinoma zastarele in se redko uporabljajo. Obenem je model skladišča ravno pri izločenih kandidatih večinoma neprimeren za naše potrebe.

2.2.1.2 Model skladišča

Model skladišča je naslednji in je izjemno pomemben kriterij, s katerim bomo izločili naslednje možne programske rešitve. Želimo si take, ki omogočajo model odjemalec-strežnik. Uporabniki bodo večinoma popravljali le nekaj dokumentov na leto. Lokalna in porazdeljena skladišča tako nikakor ne pridejo v poštev.



Slika 2.3: Graf števila preostalih kandidatov, ki zadostujejo pogoju izbire modela skladišča

S pomočjo tega kriterija izbire smo izločili dodatnih 8 programskih rešitev in sicer: Bazaar, Code Co-op, darcs, Git, GNU arch, Mercurial, Monotone in SVK.

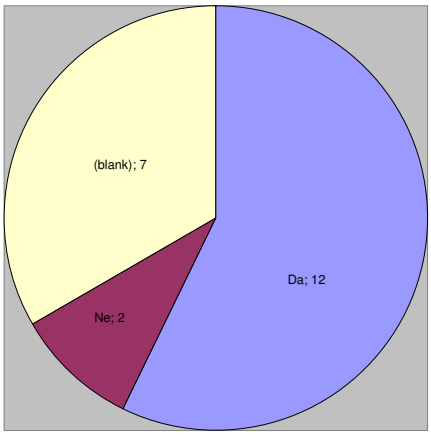
Pregled programskih rešitev, ki so nam preostale nam pokaže, da vsa orodja podpirajo strežnike na operacijskem sistemu Windows (vsaj preko Jave) in da vsa nudijo odjemalce na vseh operacijskih sistemih, za katere je mogoče, da se bodo na banki uporabljali. Zato tega kriterija ne moremo uporabiti za zmanjšanje števila kandidatov.

2.2.1.3 Stopnja zrelosti izdelka

Tretji kriterij, po katerem bomo poskušali izločiti še nekaj dodatnih neprimernih orodij, je stopnja zrelosti izdelka. Kriterij je implicitno podoben prvemu kriteriju izločanja, vendar smo takrat želeli izločiti tiste programske rešitve, ki niso uspele na tržišču, sedaj pa želimo, da nam ostane tak izdelek, ki je podprt in pri kateremu se dodajajo nove funkcionalnosti. Na podlagi tega kriterija smo dodatno izločili še CVS in Visual Source Safe. Oba izdelka sta v opuščanju in na primeru VSS je mogoče reči, da ni nikoli dejansko zaživel, saj ga za razvoj ni uporabljal niti Microsoft. CVS je že dalj časa v opuščanju in večina njegovih uporabnikov je že prešla na CVSNT ali Subversion. Obstajajo tudi konverterji, ki CVS skladišče lahko spremenijo v CVSNT ali Subversion obliko skladišča.

2.2.1.4 Večjezična podpora orodja

Četrti kriterij je na prvi pogled manj pomemben, a je kot že omenjeno pri poglavju o kriterijih izbire, glede na vrsto uporabnikov kritičen. Velika večina uporabnikov na banki ne pozna angleščine oz. jo pozna na zelo osnovni ravni. Strategija glede poimenovanja datotek ne predvideva opuščanja šumnikov in podobno. Pomembno je torej, da programska rešitev podpira večjezičnost v vsebini, odjemalcih, poimenovanju datotek ipd. Izločili bomo vsa orodja razen tistih, za katera vemo, da imajo večjezično podporo.



Slika 2.4: Razmerje med preostalimi kandidati, ki izrecno podpirajo večjezično podporo in ostalimi

S pomočjo tega kriterija smo izločili še neprimernih 9 programskih rešitev: AccuRev, Aldon, AllChange, DesignSync, LibreSource Synchronizer, PlasticSCM, Razor, SourceHaven in Vault.

2.2.1.5 Ostale tipične težave in interni mehanizmi sistema za nadzor različic

Peti kriterij izbire naj bo kriterij, s pomočjo katerega so rešene ostale tipične težave in interni mehanizmi sistema za nadzor različic. V naši razpredelnici smo zabeležil 10 od 17-ih kriterijev opisanih v prejšnjem poglavju pod to točko. V razpredelnici so nekateri aspekti in kriteriji združeni, saj so soodvisni ali izključujoči. Te kriterije smo nastavili tako, da je odgovor »Da« vedno pozitiven in predstavlja potrebno prednost. Izločili bomo vse tiste kandidate, ki ne ustrezajo dvema tretjinama te skupine kriterijev. Gre namreč za kriterije, ki se v večini primerov lahko zaobidejo z drugačno strategijo uporabe. Je pa potrebno poudariti, da tiste programske rešitve z manj realiziranimi mehanizmi zahtevajo več časa za integracijo v druge mehanizme, bolj stabilne strežnike in delovne postaje, več truda pri uporabi in celo potrebo po do-programiranju take funkcionalnosti preko katerega od skriptnih jezikov.

Število pozitivnih odgovorov / Ime	
5	
AllFusion Harvest Change Manager	
6	
Alienbrain	
SourceAnywhere Hosted	izločeni
7	ustrezni
StarTeam	
Subversion (SVN)	
Telelogic Synergy	
CVSNT	
Surround SCM	
9	
ClearCase	
Perforce	
PureCM	
10	
Team Foundation Server	

Tabela 2. 2: Seznam vseh programskih rešitev grupiranih po številu pozitivno rešenih 10-ih tipičnih težev in internih mehanizmov

Pri teh destih kriterijih bomo prešteli torej vsa polja, ki vsebujejo »Da«, zahtevamo pa, da ustrezajo vsaj sedmim (tabela 2.2). Na ta način smo dodatno izločili še 3 kandidate: Alienbrain, AllFusion Harvest Change Manager in SourceAnywhere Hosted. Pri pregledu njihovih lastnosti vidimo, da nobeden nima referenc katerega od velikih podjetij in da je primerov uporabe teh izdelkov malo.

2.2.1.6 Cena

Zadnji kriterij, s katerim bomo zožili krog najboljših kandidatov je seveda cena. Potencialnih uporabnikov te programske rešitve je 1500. V prvem koraku bo novo metodologijo verzioniranja dokumentacije takoj začelo uporabljati okoli 300 uporabnikov. Izvleček iz tabele zbranih podatkov [priloga 1] vsebuje vse podatke [tabela 2.4], ki jih potrebujemo za odločitev. Cena je pri tej skupini izdelkov precej visoka. Najnižja cena med preostalimi rešitvami je 200 ameriških dolarjev na uporabnika. Obstaja možnost nakupa konkurenčnih licenc, vendar je licenca zasedena še (navadno) 30 minut po vsaki operaciji na skladišču (tudi pregledu atributov datoteke). Projekcije normalne uporabe bi torej omogočale največ 7 uporabnikov na tako licenco, ki pa stane najmanj 4380 ameriških dolarjev. Tako visoke cene bi onemogočale širjenje te metodologije na druge projekte, saj bi se projektni vodje težko odločili za rešitve, ki jim krnijo proračun projekta. Glede na prisotnost velikega števila strokovnjakov iz področja informatike v NLB d.d. in zelo spretnih sistemskih administratorjev, je potreba po plačani rešitvi nelogična. Tako plačana rešitev, kot neplačana, zahtevata veliko truda za vzdrževanje in praktično enak obseg dela, ki je potreben za namestitev, nastavljanje, vpeljavo sistema in vzdrževanje. Izločili bomo torej vse rešitve, ki ne omogočajo brezplačne uporabe programskih rešitev. Plačana podpora je celo zaželena, saj nam dovoljuje, da se v primeru težav poslužimo poklicnih

svetovalcev. Ker imamo že na voljo strokovnjake, ki imajo čas administrirati sistem, ne želimo plačevati dodatnih stroškov licenc.

Cena / Ime
ClearCase
$4380 za konkurečno licenco + davek (vključuje 12 mesecev podpore)
CVSNT
Zastonj ali plačljivo s podporo
Perforce
Zastonj do 2 uporabnika, in za OSS razvoj; običajna cena $900 na sedež s popusti na količino
PureCM
Zastonj za 2 uporabnika. $1,000 za 5 uporabnikov
StarTeam
$7500 za konkurečno, $2500 za poimensko licenco. Možnost popusta z direkten kontaktom s podjetjem Borland
Subversion (SVN)
Zastonj (Plačljiva podpora in storitve so na voljo)
Surround SCM
Plačljivo
Team Foundation Server
Licencirano preko MSDN naročnine ali neposrednega nakupa (individualna ponudba)
Telelogic Synergy
Ponudba na individualni ravni: kontakt Telelogic

Tabela 2.3: Podatek o možnostih pridobitve programske rešitve (cena)

Na ta način smo izločili naslednje programske rešitve: ClearCase, Perforce, PureCM, StarTeam, Surround SCM, Team Foundation Server in Telelogic Synergy. Gre za skupino izjemno priznanih aplikacij, ki predstavljajo hrbtenico razvoja večine aplikacij, ki jih pišejo velike multinacionalke. Predvsem ClearCase je zaradi svoje robustnosti in mnogih spremnih orodij uporabljan pri razvoju na podjetjih HP, IBM, Cisco, Motorola, Siemens, Ericsson, Nokia in še mnogih drugih. ClearCase ima razvit lastni datotečni sistem mvfs, ki je ga odlikuje hitrost in uporabnost pri avtomatizaciji pakirnih skript izdelkov na operacijskih sistemih UNIX [3,25].

TeamFundationSystem uporablja Microsoft za svoj razvoj. Perforce je poznan vsem tistim, ki so se kadarkoli ukvarjali s Perlom in FreeBSDjem[3]. Vendar ravno te reference kažejo na to, da so orodja, ki so izpadla, neprimerna za našo ciljno uporabo. Večina izpadlih orodij se uporablja za razvoj in spremljanje izvorne kode. Največja prednost teh programskih rešitev je »multisite« razvojni model z zrcaljenjem skladišč. Orodja, ki so priložena, večinoma služijo za »datamining« in za pomoč pri doslednem označevanju, ki služi za označevanje večjih izdaj programske opreme. Ta skupina orodij ne pozna ničesar

močnejšega kot so ClearCaseovi config-speci. Ti natančno opredeljujejo načine prevzemanja in lahko samodejno vejajo pri prevzemu datoteke in podobne funkcije s pomočjo preproste sintakse.

2.2.2 Končna primerjava in izbira med ustreznima orodjima CVSNT in SubVersion

V tem trenutku sta nam ostali le še dve programski rešitvi, ki jih nadalje lahko obravnavamo. To sta CVSNT in SubVersion. Spremnih orodij, ki podpirajo eno ali drugo rešitev je približno enako. Obe rešitvi sta brezplačni in imata možnost plačljive strokovne podpore. Glede referenc je nemogoče reči, katera se je skozi čas izkazala za bolj trpežno. SubVersion se pod svojo referenčno listo ponaša s ASF, SourceForge, Google Code, KDE, GNOME, GCC, Ruby, Python, Mono, PuTTY, Zope, Xiph, GnuPG, CUPS, Wireshark, TWiki, Django in mnogo drugih, predvsem programskih podjetij. CVSNT je prav tako prisoten pri skoraj vseh podjetjih, ki se uvrščajo v lestvico Fortune 500. Pri slednjem torej lahko rečemo, da ima bolj heterogeno oz. raznovrstno paleto podjetjih, ki ga uporabljajo. Najverjetneje zaradi tega, ker vsa literatura aplikaciji CVSNT priznava, da je defakto novi standard sistemov za nadzor različic. Poleg tega bolj natančen pregled ugotavlja nekatere tehnološke dovršenosti, ki jih ima CVSNT pred SVN.

Poglejmo zgolj razlike med obema aplikacijama in sicer najprej glede na to, kaj ima CVSNT česar SVN nima. CVSNT ima glede nivoja zrelosti strežnika nekaj prednosti, ki že lahko zadostujejo, da naša odločitev pretehta na stran CVSNT. Podpira namreč identifikacijo uporabnika preko Microsoft Active Directory. Ravno tako je strežnik mogoče opravljati preko Windows nadzorne plošče. Za potrebe administracije je to premalo, pa vendar je prednost, da je mogoče strežnik na hitro ustaviti ali ga ponovno zagnati. CVSNT ima tudi bistveno večji nabor protokolov, preko katerega deluje. Najpomembneje je, da lahko deluje preko Microsoft protokola SSPI, ki omogoča aplikaciji uporabo različnih varnostnih protokolov in modelov na mreži ali računalniku tako, da se uporabniku ni potrebno ponovno identificirati. [8] Med časom delovanja je mogoče kateregakoli od protokolov, ki jih CVSNT uporablja izključiti in tako onemogočiti dostop odjemalcem, ki želijo preko teh protokolov dostopati. CVSNT deluje v native načinu NTFS, bere lahko pravice datotečnega sistema in preko teh pravic določa dostopne pravice na skladišču. Strežniška aplikacija deluje in je prevedena v UTF8 znakovnem naboru. Sistem polno podpira vse datoteke v obliki Unicode. Namestitvene datoteke za strežnik so na voljo tudi v Windows .msi obliki, kar lajša avtomatizirano posodabljanje strežnikov[27].

Družina sposobnosti	Opis	CVSNT	SVN
Funkcionalnost	Podpira avtentifikacijo preko Microsoft Active Directory ali SSH (le za Windows)	**DA**	NE
Funkcionalnost	Podprta večina novejših ukazov CVS	**DA**	**DA**
Funkcionalnost	Preimenovanja so spremljana	**DA**	**DA**
Funkcionalnost	Metapodatki o datoteki so hranjeni	**DA**	**DA**
Funkcionalnost	Objave so resnično atomske	**DA**	**DA**
Funkcionalnost	Ne zahteva samostojnega strežnika Apache	**DA**	**DA**
Funkcionalnost	Je lahko tuneliran preko SSH	**DA**	**DA**
Funkcionalnost	Učinkovito shranjevanje binarne datoteke s pomočjo binarne delte	**DA**	**DA**
Funkcionalnost	Možnost zaklepanja (rezerviran prevzem).	**DA**	NE
Funkcionalnost	Resnična podpora preimenovanju (ki ne temelji na akcijag kopiranje / brisanje)	**DA**	NE
Funkcionalnost	Varnost na nivoju skladišča	**DA**	NE
Funkcionalnost	možnost vtičnikov na strežniški strani za pregledovajne razlik med različicami	**DA**	NE
Funkcionalnost	Možnost nastavitve forsiranih protokolov, ki omogočijo strežniku, da onemogoči dostop odjemalcem, ki se povzujejo z ne-varnimi ali pomanjkljivimi nastavitvami.	**DA**	NE
Funkcionalnost	Preprosto odstranjevanje protokolov (brez ponovnega prevajanja)	**DA**	NE
Funkcionalnost	ACL na veji se lahko uporabi za omejevanje dostop	**DA**	NE
Funkcionalnost	Bolj napredni ali dodatni sprožilci so na voljo na primer po-objavljanje. Sprožilci na voljo tudi preko vmesnikov COM / DLL / .so	**DA**	NE
Funkcionalnost	Podpira Unicode datoteke, ki vsebujejo ključne besede	**DA**	NE
Funkcionalnost	Strežniške privzete možnosti (cvsrc)	**DA**	NE
Funkcionalnost	UTF-8 strežnik	**DA**	NE
Funkcionalnost	Večjezična podpora za datoteke	**DA**	**DA**
Funkcionalnost	Podpora za Rendevous	**DA**	NE
Funkcionalnost	Strežniška aplikacija za operacijske sisteme Windows, Mac OS X, Linux, Solaris	**DA**	**DA**
Funkcionalnost	Strežniška aplikacija za HPUX	**DA**	NE
Funkcionalnost	Podpora za odjemalce IBM iSeries (AS/400) OS/400	**DA**	NE
Windows strežnik	Nastavljivo iz Windows *Nadzorne plošče* Windows	**DA**	NE
Windows strežnik	Združljiv z NTFS ACL za uporabo omejevanja dostopa preko uporabniškega imena in skupine Windows	**DA**	NE
Windows strežnik	Sprožilci na voljo tudi preko COM in DLL vmesnikov	**DA**	NE
Windows strežnik	Uporablja dostop do datoteke operacijskega sistema	**DA**	NE
Windows strežnik	Možnost razhroščevanja crashdump-ov (datoteka s popolno vsebino spomina ob prenehanju delovanja aplikacije)	**DA**	NE
Windows strežnik	Windows .msi paket za nameščanje	**DA**	NE
Odjemalec	Pametno zlivanje (sledenje zlivanju) z uporabo točk zlivanja	**DA**	NE
Odjemalec	Podpira Unicode datoteke, ki vsebujejo ključne besede	**DA**	NE

Tabela 2.4: Primerjava CVSNT in SVN [27]

Vsega zgoraj naštetega SubVersion nima oz. ne zmore. Možno je doseči enake funkcionalnosti, vendar je za to potrebno integrirati posebna orodja, s katerimi dosežemo enak učinek. Edina bistvena prednost SVN pred CVSNT je, da je preimenovanje datotek veliko bolj stabilno in ga je mogoče uporabljati iz TortoiseSVN odjemalca.

Poleg tega pa je bilo mogoče na podjetju Hermes Softlab najti več strokovnjakov z izkušnjami iz CVSNT. S pomočjo kadrovske baze znanja je CVSNT premagal SubVersion s skoraj dvakrat več zaposlenimi, ki imajo poglobljene izkušnje z njim [26].

2.3 Izbrano orodje in način povezave

Izbrali smo torej aplikacijo CVSNT. Namestitev bomo razdelili na dva dela. Strežniško namestitev in namestitev na delovne postaje uporabnikov.

Strežnik podatke shranjuje na datotečnem sistemu v obliki datotek. Znotraj datotek uporablja svojo strategijo označevanja sprememb in beleženja zgodovine. Zahteve po strojni opremi so za CVSNT strežnik veliko manjše kot za sam operacijski sistem. Pogoji za strojno opremo so torej pogojeni bolj od operacijskega sistema kot strežniške aplikacije. S povečanjem števila uporabnikov zelo počasi narašča le potreba po dodatnem spominu. Podprti sistemi za strežnik so vsi Microsoftovi operacijski sistemi, vendar je zaradi vsesplošne varnosti bolje izbrati enega izmed »high end« strežniških rešitev. Naš novi strežnik CVSNT bo tekel na računalniku z operacijskim sistemom Windows 2003 Advanced Server. V nadaljevanju (2.3.1) bomo pregledali, kaj vse mora biti nameščeno na strežniku poleg same strežniške aplikacije.

Zelen kvadrat z belo kljukico	
	Na datoteki: Datoteka je objavljena. Na mapi: Vse datoteke v mapi so že objavljene.
Oranžen kvadrat z belo puščico v levo:	
	Na datoteki: Datoteka je urejena in čaka na objavo. Na mapi: Mapa vsebuje datoteke, ki čakajo na objavo.
Oranžen kvadrat z belim plusom:	
	Na datoteki: Datoteka je dodana in čaka na prvo objavo. Na mapi: Mapa vsebuje dodane datoteke, ki čakajo na prvo objavo.
Svetlomoder kvadrat s ključavnico:	
	Na datoteki: Datoteka je spremljana. Za spremembe jo je potrebno najprej »urediti«. Na mapi: Mapa vsebuje datoteke, ki so spremljane.
Moder kvadrat z belim vprašajem:	
	Na datoteki: Datoteka se ni pod nadzorom različice in ni niti prezrta. Na mapi: Mapa še ni pod nadzorom različice in ni niti prezrta.
Rdeč kvadrat z belim klicajem:	
	Na datoteki: Obstajajo neskladnosti med datoteko in skladiščem. Na mapi: V mapi obstajajo datoteke z neskladnostmi ali obstajajo neskladnosti med mapo in skladiščem.
Siv kvadrat z belim križcem:	
	Na datoteki: Datoteka je prezrta in ni pod nadzorom različice. Dostopna je samo v tem peskovniku. Na mapi: Mapa je prezrta in ni pod nadzorom različice. Dostopna je samo v tem peskovniku.

Tabela 2.5: Ikone odjemalca TortoiseCVS [9]

Odjemalec ravno tako deluje na vseh Microsoftovih platformah in za delovanje potrebuje zelo malo strojnih virov. Med vsemi odjemalci bomo izbrali najbolj razširjenega, ki ima tudi možnost integracije v Windows Explorer. To je TortoiseCVS. Odjemalci so na voljo tudi na drugih platformah (UNIX, Linux), vendar večinoma delujejo samo iz ukazne vrstice, le redko imajo lasten GUI. TortoiseCVS je v svoji osnovi „wrapper“ preko ukazov iz ukazne vrstice, vendar ima tudi druge zmožnosti (grafičen prikaz drevesa revizije ipd.). TortoiseCVS nam torej z miško prek grafičnega vmesnika omogoča upravljanje s CVSNT

nadzorovanimi datotekami. TortoiseCVS tudi označuje stanje datotek tako, da čez privzete ikone datotek in map izriše manjši znak, ki grafično prikaže stanje, v katerem se ta objekt nahaja. Ta dodatni znakec je viden v spodnjem desnem kotu originalne ikone in je v veliko pomoč uporabnikom, da lažje opazijo stanje datotek v neki mapi.

2.3.1 Namestitev strežnika

Namestili smo CVSNT strežnik za potrebe projektov Globus in NBO. Sama nastavitev strežnika je neodvisna od projekta. So pa na strežniku potrebne še dodatne aplikacije in interpreterji. Ti dodatki bodo omogočili obveščanje po elektronski pošti, poganjanje samodejnih vzdrževalnih skript in grafičen pregled preko spletnega vmesnika. Dodana je tudi aplikacija, ki omogoča grafično nastavljanje filtrov za pregledovanje zgodovine skladišča.

Izbrali smo Windows Advanced Server 2003 operacijski sistem. Nanj smo namestili tudi Internet Information Services 6.0 (IIS6). Sproženo je bilo posodabljanje obojega in s tem namestitev zadnjih priporočenih popravkov. Naknadno posodabljanje operacijskega sistema ne bi smelo vplivati na kasneje nameščene aplikacije. Seveda je vedno dobro narediti varnostno kopijo celotnega sistema pred posodabljanjem in po vsakem posodabljanju sistema vsaj grobo preveriti glavne funkcionalnosti vsakega od nameščenih orodij.

Strežniške aplikacije so bile nameščene v tem vrstnem redu:

1. CVSNT strežnik (strežnik datotek)
2. ActivePerl paket (podpora za vzdrževalne skripte)
3. ActivePython paket (podpora za skripte ViewCVSja)
4. CVSMailer aplikacija (podpora za obveščanje o spremembah po elektronski pošti)
5. ViewCVS strežnik (pregledovanje datotek s pomočjo spletnega brskalnika)
6. cvshistory (pregledovanje zgodovine skladišča preko spletnega brskalnika – podpora za poročila)
7. WinMerge (samodejno zlivanje sprememb)
8. metadiff (pregledovanje razlik med datotekami glede na tip datoteke)

Prijava na strežnik lahko poteka preko protokolov SSPI, EXT, SSH ali PSERVER, ki jih podpirata tako strežnik kot odjemalec. V NLB je za komunikacijo uporabljen samo protokol SSPI. Že pri sami namestitvi strežnika izberemo kot edini protokol SSPI. SSPI protokol izvede identifikacijo uporabnika neposredno na domenski krmilnik. SSPI ne pošilja uporabniškega imena in gesla preko mreže, ampak samo potrdi žeton, ki se prenese na strežniško stran. SSPI (Security Support Provider Interface) je Microsoftova različica protokola GSSAPI (Generic Security Service API), ki abstrahira prenosni protokol in ga loči od same aplikacije. To je realizirano v dinamičnih knjižnicah, ki se imenujejo SSP in so ločeni od strežnika ali odjemalca. Uporabniku torej ni potrebno izkazovati svojega uporabniškega imena ali gesla. Vse se izvede samodejno z grafičnim odjemalcem TortoiseCVS [11,12,19]. Vnos uporabniškega imena in gesla je potreben samo za delo v ukazni vrstici.

CVSNT strežnik zagotavlja več nivojev nastavljanja pravic in nadzora nad modulom, mapo in drevesom različic. V sodelovanju z datotečnim sistemom NTFS je mogoče omejiti tudi dostop do posamezne datoteke.

Vse datoteke potrebne za administracijo skladišča se nahajajo v mapi »CVSROOT«. Ta mapa mora imeti nastavljene pravice tako, da jo lahko prevzamejo le skrbniki in ne navadni uporabniki. Navadnim uporabnikom naj bo preprečeno branje in pisanje v to mapo. Tako je prevzem onemogočen vsem, razen pooblaščenim uporabnikom [11,12].

V tem trenutku lahko za nekaj časa pustimo strežnik pri miru in namestimo še vse ostale pakete in aplikacije. Namestitev interpreterjev Perl in Phyton je večinoma enostavna. Izberemo, kam jih bomo namestili. Glede na to, da sta oba interpreterja potrebna za poganjanje ostalih strežniških aplikacij, je dovolj izbrati konzervativno obliko namestitve, ki ne vsebuje urejevalnikov, ali katerih drugih dodatkov. CVSMailer, ViewCVS, cvshistory, WinMerge in metadiff so ravno tako preprosti za namestitev na strežnik. Vsakega od njih smo brez težav namestili na strežnik po korakih opisanih v njihovi spremni dokumentaciji [12,14,15].

Strežnik je polno nameščen in potrebno je dodati vsebino in določiti pravice do dostopov. Lahko začnemo vpisovati uporabnike, ki bodo uporabljali nov sistem za shranjevanje različic. Uporabniki so našteti v datoteki »users« po v naprej določeni sintaksi vpisa.

Tako vpisane uporabnike lahko naprej razporedimo po skupinah, da bi tako lažje nadzorovali dostop, pisanje, spreminjanje pravic itd. To storimo preko datoteke »group«.

Administratorjev ne definiramo znotraj datoteke »group«, ampak je za te namene na voljo ločena datoteka »admin«. Administratorji vpisani v to datoteko imajo pravico vedno, neglede na trenutne nastavljene pravice kateregakoli elementa, spreminjati pravice in lastnike vej, map in različic. Ravno tako samo ti uporabniki lahko spreminjajo mapo \CVSROOT in s tem nastavitve CVSNT strežnika[11].

Ko smo vse uporabnike vpisali v te datoteke, lahko začnemo določati pravice za posamezne mape, veje ali različice. Pred tem moramo ustvariti prve module in objaviti prve, začetne različice datotek. Nad njimi bomo z zaporedji ukazov za določanje lastnika in določanje pravic ustvarili strukturirano drevo, ki bo ponazarjalo ureditev znotraj skupin uporabnikov in njihovega dela na projektu. Nastavitve o tem, kdo ima katere pravice, so odvisne od odločitvene strukture projekta in sprotne presoje o poslovnih skrivnostih v nastajajoči dokumentaciji. Pred prvim prevzemom smo stare mape že strukturirano uredili po funkcijskih skupinah poslovnih uporabnikov, modulih aplikacij, ki so razvijane pri projektih in ostalih možnih ureditvah datotek, po v naprej določeni strategiji. Stare in odslužene datoteke smo shranili v pod-mape “ARHIV”, da bi preprečili morebitno izgubo kakega pomembnega podatka. Začetna ideja, da bi mape preuredili tako, da bi staro stanje že naložili v začetno skladišče, se je izkazala za nepotrebno izgubo energije. Stare datoteke so bile preveč različno poimenovane in številčno jih je bilo preveč . Zato smo le imena zadnjih različic datotek očistili datumov (kjer ne gre za zapisnike sestankov na določen dan) in raznoraznih označb revizij in avtorstev.

Strežnik je polno delujoč in pripravljen za uporabo. Potrebno je urediti še skupni prevzem, preko katerega bodo dostopali vsi uporabniki. Sistem za nadzor hkratnih različic je največkrat v uporabi pri geografsko razpršenih razvijalcih. Zaradi tega je v navadi, da vsak uporabnik s prevzemom ustvari lastno lokalno kopijo (svoj peskovnik) določenega modula. Naknadno le še posodablja svojo kopijo in jo objavlja na strežniku. Ta strategija je odlična,

če želimo zmanjšati čas za dostop do datotek pri oddaljenem uporabniku in mrežni promet. Toda zavedati se moramo, da je potrošnja diskovnega prostora enaka zmnožku števila uporabnikov s potrebnim prostorom za zadnjo različico datotek. V okolju, kjer se vsi lokalni podatki tudi varnostno shranjujejo, je to nesprejemljivo, saj je redundanca varnostnih kopij prevelika (število enakih datotek je približno enako številu uporabnikov). Razpršeni uporabniki nimajo enostavnih možnosti, da bi imeli nek skupen diskovni prostor brez varnostnih pomislekov (razen, če so znotraj iste mreže, VPN itd.). Ker pa so v NLB vsi uporabniki dokumentacije povezani v hitro omrežje in prijavljeni v domeno, se naravno ponuja možnost uporabe skupnega prevzema. Glede na to, da se ta skupni prevzem nahaja na particiji NTFS, je tu podpora CVSNT nekoliko okrnjena in je potrebno vsem odjemalcem določiti nastavitve, ki so drugačne od privzetih. S takim načinom prevzema se dojemanje skupnih datotek ne bo bistveno spremenilo glede na prejšnji način dela in uporabnikom bo omogočen mehek prehod na novi način dela. Posebna pozornost je potrebna le s strani administratorja, ki izvaja nastavitve sistema, saj mora biti pozoren na to, da so vsi prevzemi in ustvarjanje peskovnika brez uporabniška imena in generični. CVSNT v tem primeru, ob vsakem poskusu dostopa do skladišča, vedno sproti znova identificira uporabnika.

2.3.2 Namestitev odjemalcev

TortoiseCVS je odjemalec za datoteke, ki so pod nadzorom verzije na strežniku CVSNT. Take datoteke smo v preteklosti morali upravljati iz ukazne vrstice. TortoiseCVS nam omogoča delo z miško prek grafičnega vmesnika, kar rabo zelo olajša. TortoiseCVS je grafični vmesnik do ukaza »cvs«. Uporabniku prihrani poznavanje ukazov in njihovih parametrov. V grafičnem vmesniku izberemo vse možnosti, ki nam jih ponuja in tako TortoiseCVS sestavi ukazno vrstico, ki jo bo samodejno izvedel. TortoiseCVS bo tudi prestregel vse odgovore tega ukaza in jih prikazal v grafičnem načinu [9,14].

TortoiseCVS omogoča, da še vedno lahko vse operacije izvedemo tudi iz ukazne vrstice. V tem primeru moramo imeti pravilno nastavljeni $PATH in $CVSROOT spremenljivki okolja. Zahtevnejše administrativne operacije niso podprte v grafičnem načinu in jih je potrebno v ukazni vrstici izvesti ročno. Če želimo preimenovati ali premikati datoteko, je smiselno poklicati administratorja CVSNT. To pomanjkljivost bomo odpravili s tem, da bomo to izpostavili v tehnični in uporabniški dokumentaciji in uporabnike tudi preko delavnic izobrazili v tej smeri, da vedo, katerih operacij ne smejo izvajati sami. Te funkcionalnosti bomo v odjemalcu onemogočili in tudi tako uporabnike prisili, da se držijo pravil.

Namestitve odjemalcev na delovne postaje opravlja PC Team preko orodij SMS in ScriptLogic. Namestitev je avtomatizirana in je sestavljena iz dveh delov: namestitve .msi paketa odjemalca in nastavitev glede na projekt, pri katerem je uporabnik sodeluje.

Prvi del namestitve odjemalcev je vključitev .msi paketa z distribucijo TortoiseCVSja v SMS seznam paketov, ki se distribuirajo na delovne postaje. Za potrebe tega je ustvarjena domenska skupina, ki vključuje vsa domenska imena delovnih postaj, na katere je potrebno TortoiseCVS namestiti. SMS bo torej na vse delovne postaje namestiti aplikacijo TortoiseCVS s privzetimi nastavitvami (tabela 2.2).

Drugi del namestitve obsega natančne nastavitve odjemalcev, ki bodo dostopali do datotek katerega od projektov. TortoiseCVS zapisuje svoje nastavitve v register operacijskega sistema Windows in prikazuje možnosti v meniju ob kliku desne tipke v oknu Windows Explorer iz datoteke tortoisecvs.config. Zaradi težav z večjezičnostjo TortoiseCVS bomo v ta del namestitve vključili tudi na novo prevedeno datoteko tortoisecvs.mo (prevedeno v UTF-9 naboru znakov, saj lokalizacija v originalni distribuciji ni pravilno prevedena – že nekaj let). Ustvarili bomo domenske skupine, ki bodo vsebovale vsa domenska uporabniška imena uporabnikov, ki so vključeni v določen projekt (tabela 2.2). Preko orodja ScriptLogic bomo poskrbeli za:

- vpis nastavitev v register operacijskega sistema Windows in tako omogočili podporo za prevzeme na skupnih omrežnih virih in kazalce na orodju za primerjanje datotek ter samodejno zlivanje (metadiff in WinMerge)
- zamenjavo kataloga sporočil v slovenščini v naboru znakov UTF-9 in tako omogočili pravilen izpis šumnikov
- zamenjavo menijskih izpisov na desnem kliku in tako onemogočili dostop do naprednih funkcij, ki bi lahko povzročile težave
- določili bomo nove $PATH in $CVSROOT spremenljivke okolja in tako omogočili delo tudi iz ukazne vrstice
- ScriptLogic se zažene ob vsaki prijavi uporabnika in poskrbi, da so vse zgoraj naštete spremembe izvedene.

Domenska skupina	Obseg članstva
APLwksCVSNT	imena delovnih postaj
APLusrCVSNT_Globus	imena uporabnikov projekta Globus
APLusrCVSNT_NBO	imena uporabnikov projekta Globus
APLusrCVSNT_Adonis	imena uporabnikov projekta Adonis
APLusrCVSNT_AdonisR	imena uporabnikov projekta Adonis (read-only)
APLusrCVSNT_PPT	imena uporabnikov projekta Plačilni Promet s Tujino
APLusrCVSNT_UCIT	imena uporabnikov oddelka UCIT

Tabela 2.6: Imena domenskih skupin za distribucijo namestitev in nastavitev

2.4 Izdelava spremne dokumentacije

Celotno okolje je pripravljeno. Strežnik je delujoč in vsi uporabniki imajo na svojih delovnih postajah nameščene nastavljene odjemalce, vendar je opravljen komaj manjši del dela. Množico uporabnikov je potrebno uvesti v nov način dela in nova pravila skupinskega dela na omrežnih virih. Za potrebe ostalih administratorjev je potrebno izdelati natančno dokumentacijo, s katero se je mogoče spoprijeti s težavami in nastavitvami novega sistema in transparentno dodajati in odvzemati funkcionalnosti in uporabnike.

To dokumentacijo na delimo na:

- dokumentacijo za administriranje in podporo uporabnikom
- dokumentacijo za delo uporabnikov
- dokumentacijo za izobraževanje uporabnikov

2.4.1 Administratorska dokumentacija

Je izjemno pomembna, saj zaradi odnosa naročnik-podizvajalec in redne migracije kadrov, zagotavlja prenosljivost znanja in kontinuiteto tudi ob menjavi kadra. Zelo pomembno je, da tako dokumentacijo obdržimo živo in poskrbimo, da se novosti ali spremembe v postopkih redno vpisujejo vanjo. Naloga administratorja, z vidika zapisovanja tega kar je nadgradil v svojem znanju, je izredno pomembna.

Začetni dokument, ki ga je potrebno pripraviti je dokument UCIT »**Tehnično navodilo za CVSNT«**, ki že vsebuje vnaprej določena poglavja, ki jih je potrebno pokriti. Ta so:

1. Opis rešitve in tehničnih zahtev
2. Arhitektura sistema (konfiguracija strojne, programske in komunikacijske opreme)
3. Tehnološko-varnostne nastavitve sistema
4. Navodilo za namestitev (instalacijo)
5. Opis skrbniškega sistema in postopkov v primeru zunanjih izvajalcev
6. Navodilo za obratovanje
7. Navodilo za izvajanje postopkov obnovitve dela po nesreči
8. Navodilo za odpravljanje motenj (težav) in prijavo okvar
9. Navodilo za dodajanje novega uporabnika - CAU
10. Opis programskih kontrol v izdelanem izdelku

Ta dokument služi kot zbirni ali pregledni dokument drugim, bolj natančnim dokumentom. Ti so pretežno navodila, ki opisujejo postopke v predvidljivih scenarijih delovanja. Vsa dokumentacija je živa in se sproti dopolnjuje in nadgrajuje. Reference tega dokumenta obsegajo naslednje dokumente, ki jih je bilo ravno tako potrebno izdelati:

- **Administracija CVSNT** opisuje strategije, načela dobre prakse in ravnanja ob različnih dogodkih na strežniku. Referencira izseke iz originalne administratorske dokumentacije orodja CVSNT [11] in podrobno knjigo z opisom delovanja in načel dobrih praks na primeru razvoja odprtokodnih rešitev [12]
- **Namestitev CVSNT** je podroben dokument z zaslonskimi slikami, ki prikazuje namestitev CVSNT strežnika korak-za-korakom.
- **Namestitev CVSMailerja** je podroben dokument z zaslonskimi slikami, ki prikazuje namestitev orodja CVSMailer.
- **Namestitev ViewCVSja** je podroben dokument z zaslonskimi slikami, ki opisuje namestitev orodja ViewCVS
- **Namestitev cvshistoryja** je podroben dokument z zaslonskimi slikami, ki opisuje namestitev orodja cvshistory
- **Namestitev in ročno nastavljanje odjemalca TortoiseCVS** je podroben dokument z zaslonskimi slikami in opisom namestitve odjemalca na delovno postajo in nastavljanje za uporabnika
- **Izdelava statistik in poročil** je dokument, ki opisuje, kako s pomočjo orodja cvshistory izdelamo tedensko poročilo aktivnosti na dokumentaciji projekta.

2.4.2 Uporabniška dokumentacija

V vsakem velikem podjetju je prihod novega kadra zelo običajen pojav. Ti imajo na voljo uvajalno obdobje, v katerem morajo večinoma sami preučiti veliko število novih aplikacij

in postopkov. Dobro oblikovana uporabniška dokumentacija nam prihrani veliko časa, obenem pa pomaga tudi starim uporabnikom, ki nekaterih orodij ne uporabljajo redno, da si osvežijo znanje. Na ta način razbremenjujemo centre za pomoč uporabnikom, ki prepogosto odgovarjajo na najbolj osnovna vprašanja. V načelu z dobro prakso je uporabnika napotiti do pravega dokumenta, saj bo na ta način naslednjič poskušal najti odgovor sam.

Dokumenti, ki jih je bilo potrebno izdelati za zadovoljitev potreb uporabnikov so:

- **Navodila uporabe TortoiseCVS** je dokument, ki obsega
  1. Principi CVSNT strežnika in odjemalca TortoiseCVS
     - Razlaga revizije, oznake in veje datoteke
     - Razlaga zgodovine datoteke
  2. Scenariji uporabe
     - Scenarij, ki se uporablja za dokumentacijo projekta GLOBUS
     - Scenarij, ki se uporablja za dokumentacijo projekta NBO
  3. Uporaba odjemalca
     - Desni klik v »Windows Explorerju«
       $\Rightarrow$ Ikone
       $\Rightarrow$ Ukazi
     - Delo z ukazno vrstico
  4. Ključne besede v tekstovnih datotekah
  5. Osnovni pojmi
- **CVSNT in TortoiseCVS pogosta vprašanja** je dokument, ki vsebuje pogosto zastavljena vprašanja uporabnikov in odgovore na le-ta. Dokument živi in administratorji CVSNT sproti vpisujejo vanj novo prispela vprašanja in odgovore, ki so jih uporabnikom dali.
- **Spajanja Wordovih datotek** je dokument, ki vsebuje opis in zaslonske slike spajanja dveh različic istega dokumenta v novi dokument. Glede na to, da je to poglavitna in za uporabnike zelo pomembna funkcionalnost, je podrobno razložena in ima zato lasten dokument.
- **Plonklistek CVSNT** je tabela z razloženimi pojmi CVSNT. Oblikovana je tako, da se natisne in pripne poleg monitorja in je uporabniku v hitro pomoč ali osvežitev znanja, ko potrebuje uporabljati CVSNT po daljši neaktivnosti.

2.4.3 Dokumentacija za delavnice

Ob vsaki vpeljavi novega sistema v katerokoli okolje je potrebno izobraziti uporabnike, ki ga bodo uporabljali. Uspešno izdelana strategija, kaj želimo uporabnikom predstaviti, je izjemnega pomena. Dokumentacija povezana z delom na delavnici mora imeti jasno rdečo nit. Prezentacija mora biti oblikovana jasno in vsebovati kratka sporočila, ki jih potem v spremnem govoru dopolnimo s podrobno razlago.

Dodatna dokumentacija za delavnice je le še prezentacija. Ostale dokumente imamo že izdelane, saj bomo na delavnicah pretežno razlagali uporabniško dokumentacijo.

Prezentacija CVSNT je prezentacija, ki podrobno obravnava naslednja vprašanja:

- **Kaj je CVSNT?**
  Kratek opis sistema za nadzor različic. Poudarek je na prednostih, ki jih uporabnik čuti ob uporabi novega sistema. Izpostavljeno je tudi, da uporabniki brez odjemalca (iz drugih projektov) še vedno lahko delajo na dokumentih in jih pregledujejo enako kot prej.
- **Kakšne so razlike napram prejšnjemu sistemu?**
  Kako dokumentacija nastaja in kako naj izboljšajo svoje navade, da jim bo novi sistem v pomoč in ne v dodatno breme.
- **Kako ga uporabljam?**
  Natančen opis točno določenega scenarija, ki bo najbolj pogost. Popravljanje novega dokumenta, pregled zgodovine, pregled razlike napram prejšnji različici dokumenta, objava in pregled, kako se je ta objava odrazila v zgodovini datoteke.
- **Delavno navodilo za projekt.**
  Je najbolj pomemben del prezentacije. Vsebuje jasna in kratka navodila, kako se mora postopati z dokumentacijo projekta.
- **Vaje**
  Del, ki opisuje, kakšno vajo bomo naredili skupaj.

2.5 Delavnice za uporabnike

Izvesti je potrebno večje število delavnic za uporabnike. V delavnicah poskušamo ohraniti obvladljivo število slušateljev. Obvezno je, da ima vsak udeleženec svojo delovno postajo, v nasprotnem primeru bo tak udeleženec imel težave pri delu in bo morda administratorju povzročil še kakšno nevšečnost. Samozavestna in učinkovita uporaba sistema za nadzor različic je lahko dosežena le tako, da je uporabnik že uporabil sistem in svoje pomisleke razjasnil z izkušenim predavateljem. Če slušateljevih pomislekov nismo raziskali, tvegamo, da bo ta začel voditi dokumentacijo drugje in jo bo le občasno odložil tja, kjer mora biti. To je z vidika varnosti in varnostnih kopij, ki preprečujejo izgubo podatkov še slabše, kot če bi svojo dokumentacijo shranjeval na omrežnem viru, na star način.

Za vsako delavnico je potrebno izdelati:

- seznam prisotnih na delavnici, s katerim kasneje pregledujemo, kdo zna sistem uporabljati in kdo še mora biti izobražen,
- papirni izpis prosojnic prezentacije **Prezentacija CVSNT**, da si lahko udeleženci zapisujejo svoje komentarje,
- papirni izpis **Navodil uporabe TortoiseCVS.** Slušatelje opozorimo, kje na omrežnem viru ga lahko vedno dobijo v najbolj ažurni različici. Dokument namreč podrobno obravnava vse funkcionalnosti odjemalca sistema za nadzor različice in je primeren za vse vrste uporabnikov. Ta dokument smo že opisali v prejšnjem poglavju uporabniške dokumentacije.

Sam potek delavnice je odvisen od predavatelja do predavatelja. Po mojih izkušnjah je najbolje slušatelje neprestano vključevati v temo, jih spraševati o njihovih izkušnjah in poskušati biti predvsem svetovalen. Nikakor ni dobro odpredavati svojega področja in na koncu dati možnost za vprašanja, saj na tak način slušatelj preneha s spremljanjem takoj, ko se s prvo trditvijo ne strinja. Vsak viden pomislek ali medsebojni pogovor je dobro izkoristiti za sproščen pogovor o tem, kako mu bomo lahko neko zadevo olajšali s pomočjo

novega sistema. Nov navdušen uporabnik je tudi najboljši ambasador novega sistema med ostalimi zaposlenimi. S sistemi je namreč tako, da se jih morajo držati in jih uporabljati vsi, če želimo, da so učinkoviti in njihove vrline pridejo do izraza. Zaposleni morajo poznati in začeti uporabljati nove pojme, ki izhajajo iz uporabe nove rešitve za nadzor različic.

Najpomembnejši odsek pri delavnici CVSNT je tisti, ki obravnava dogovor na nivoju projekta. To je skupek pravil, ki so nujno potrebna za pravilno delovanje sistema.

2.5.1 Scenarij, ki se uporablja za dokumentacijo projekta Globus:

Scenarij uporabe orodja CVSNT je prilagojen specifičnim potrebam projekta Globus.
Navodilo, ki je bilo predano vsem članom projekta Globus:

- Prevzem na skupnih diskih zahteva, da so pravice dostopa na ravni datotečnega sistema usklajene s pravicami dostopa v CVSNT. Nihče ne sme prevzeti ničesar s svojim lastnim uporabniškim imenom. Prevzema lahko le administrator CVSNT. Peskovnik se prevzame z neizpolnjenim poljem »user«. Tako dosežemo, da se ob vsaki operaciji na strežniku preveri identiteta uporabnika.
- Datotek, ki so označene kot spremenjene in jih je spreminjal nekdo drug, ne spreminjamo. Njihov lastnik jih mora objaviti in šele nato jih spreminjajmo-urejamo. Tako bomo preprečili, da spreminjamo delovno verzijo, ki jo pravkar spreminja nekdo drug. Ker običajna orodja (npr. Word) pri urejanju zaklepajo dokumente, je možnost, da nehote spremenimo že urejani dokument, majhna. V primeru, da bo želel uporabnik pridobiti svoj dokument, ki ga trenutno spreminja nekdo drug, lahko iz zgodovine datoteke pridobi natanko to različico z ukazom »lepljivo«. V primeru, da ne najdete tistega, ki je urejal dokument in ga ni objavil se obrnite na administratorja CVSNT ali v **Pogostih vprašanjih** poglejte kako lahko iz NTFS vidite kdo je trenutni lastnik datoteke, do katere ne morete dostopati.
- Uporaba privatnih peskovnikov ni dopuščena. Uporaba privatnih peskovnikov bi predpostavljala, da se vsebina skupnih diskov ažurira z njimi. V primeru, da tak način potrebujete se obrnite na svojega projektnega vodjo.
- Posodabljanje imenikov se na mrežnem disku (skupnem prevzemu) ne izvaja. To opravlja administrator CVSNT zgolj, če obstajajo uporabniki, ki imajo lasten peskovnik (ne uporabljajo skupnega peskovnika iz objektivnih razlogov in jim je to bilo odobreno).
- Uporabniki lahko uporabljajo opcijo »Posodobi stanje mape« za ponoven izris ikon. Ta ukaz samo izriše ikone in ne posodablja samih datotek.
- Uporabniki ne brišejo nobene od map ali datotek. Pokličejo naj administratorja, če je taka operacija potrebna.
- Uporabniki ne premikajo map ali datotek. Pokličejo naj administratorja in ta jim bo pomagal izvesti akcijo.
- Uporabniki ne preimenujejo map ali datotek. Pokličejo naj administratorja in ta jim bo pomagal izvesti akcijo.
- CVSNT omogoča varnost na mapi, različici in veji. Varnost na nivoju datoteke omogoča NTFS, vendar le-to lahko opravlja samo administrator CVSNT. V primeru, da želi uporabnik onemogočiti dostop do katere izmed datotek, naj se obrne na administratorja CVSNT. Ravno tako naj se obrne na administratorja, če želi dodatne možnosti nadzora (opozorila, da nekdo spreminja datoteko, zaščita veje, mape itd.).
- Skupni prevzem je že montiran na vaše mrežno ime Q:.

2.5.2 Scenarij, ki se uporablja za dokumentacijo projekta NBO:

Projekt NBO ima scenarij uporabe, ki je prav tako različen od navadne uporabe CVSNT. Vsa pravila, ki veljajo za projekt Globus veljajo tudi za NBO. Bistvena razlika je, da je isto skladišče prevzeto večkrat z različnim kriterijem nalepke. Projekt NBO ob zaključku cikla razvoja vsem datotekam nalepi v naprej določeno nalepko z imenom izdaje.
Dodatna navodila, ki so bila predana članom projekta NBO so:

- Ob prehodu projekta na naslednji mejnik v razvoju morajo biti vse datoteke objavljene do dogovorjene ure. Ustvari se nov prevzem v mapi P:\NFA\Program\Projekti\ z imenom NBO-(N+1) (če je bil prejšnji npr. NBO-2, je novi NBO-3). V tej novi mapi se nahaja prevzem, ki ima zadnjo različico nastavljeno v novo vejo. Začne se uporabljati nova mapa za razvoj. Zgodovina je vidna na obeh mapah in je ista (tudi v stari mapi je vidna nova veja). Vidna različica datoteke je v »grafu revizij« nakazana z odebeljenim okvirjem različice.
- Mapa P:\NFA\Program\Projekti\Arhiv\ vsebuje stare datoteke. V to mapo ne moremo pisati. Na njej je še vedno mogoče pregledovati »graf revizij«, dnevnike ipd., le objava je onemogočena.
- Skupni prevzem je že montiran na mrežno ime P:.

2.5.3 Napredni uporabniki

V primeru, da je skupina sestavljena iz bolj naprednih uporabnikov, je smiselno preleteti pravila projekta. Namesto počasne razlage osnovnega scenarija je bolje opisati vse funkcionalnosti odjemalca TortoiseCVS in porabiti več časa za ostale možnosti CVSNT, kot so zaklepanje, funkcija uredi in vejanje. Navadni uporabniki zgodovino zgolj pregledujejo in gredo z različicami le naprej. V primeru, da se morajo vrniti na prejšnjo različico, skoraj vedno pokličejo administratorja, ki jim to pomaga izvesti. Ker so napredni uporabniki najpogosteje razvijalci, imajo večkrat potrebo po vračanju na staro različico in potrebujejo nasploh boljše poznavanje načinov označevanja, kot so nalepke, veje, interne številke revizij in podobno. V slučaju bolj zahtevnega avditorija je razlaga bolj natančneje opisana v naslednjih poglavjih:

- Posodobitev lastne kopije z zadnjo različico modula/mape.
- Dodajanje nove datoteke in/ali mape
- Konkurenčno popravljane datoteke
- Objava datoteke in/ali mape
- Preimenovanje datoteke ali mape
- Pregled drevesa različic datoteke ter možnosti in podatki v slednjem
- Pridobitev in popravljanje starejše različice datoteke
- Ogled starejše različice datoteke
- Pregled razlike med trenutno in predhodno različico datoteke
- Pregled razlike med dvema poljubnima različicama datoteke
- Prikaz novih kolon (“Revizija CVS”, “Commiter” …) v Windows Explorerju

2.6 Potrebe in funkcije vodstva projekta

Vodstvo projektov sistem CVSNT pozna, saj so bili soudeleženi pri izbiri ali bo sistem na njihov projekt vpeljan. Praktično IT poznavanje je pri vodstvu deljeno. V NLB d.d.

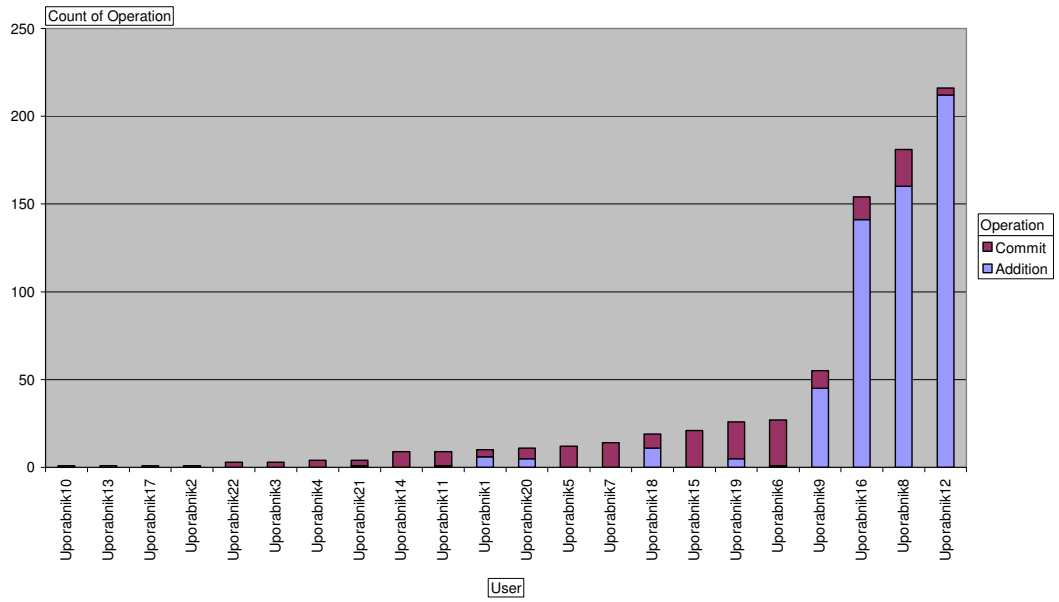
projekte vodi vedno par menedžerjev. Eden je zadolžen za poslovni del, drugi pa za IT del. V vsakem primeru morata oba vedeti, kaj jim sistem za nadzor različic nudi v pomenu uporabnosti.

Neposredno dodana vrednost za te uporabnike je:

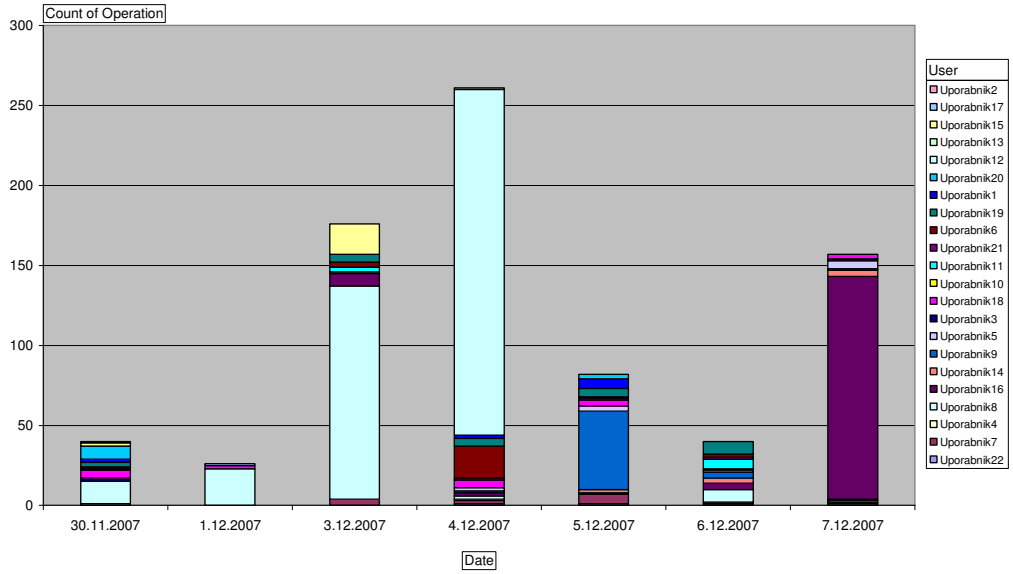
- **samodejno obveščanje skupin uporabnikov**
  V vsaki skupini uporabnikov je vključen tudi vodja projekta. Ta dobiva elektronsko pošto ob spremembi vsake od datotek. Na ta način sproti dobiva informacije, ali je kak pomemben dokument, ki ga pričakuje dokončan.
- **dostop preko spletnega vmesnika**
  Za nastope na sestankih lahko menedžer uporabi spletni vmesnik ViewCVS. Preko njega lahko pregleduje zgodovino in prikazuje drevesa revizij. Na ta način ni odvisen od namestitve odjemalca v različnih sejnih sobah.
- **poročila o aktivnosti na različnih področjih dokumentacije**
  Za potrebe vodstva se ustvari tedensko poročilo z aktivnostjo na dokumentaciji njihovega projekta. Z orodjem historycvs se izvede izpis vseh akcij na skladišču in prenese v Excel. Tam se nato lahko naredi grafe, ki večinoma prikazujejo:
    - število objav in dodajanj na uporabnika
    - aktivnosti na celotnem modulu projekta po dneh v tednu
    - aktivnost na mapah v skladišču

  To opravilo je za vodstvo tako pomembno, da je za potrebe administratorja opisano v posebnem dokumentu, opisanem v točki 3.4.1. Če je modul v skladišču pravilno organiziran z mapami in pod-mapami, je mogoče glede na nivo videti, česa je dani teden bilo več: razvoja, raziskovanja ali testiranja.

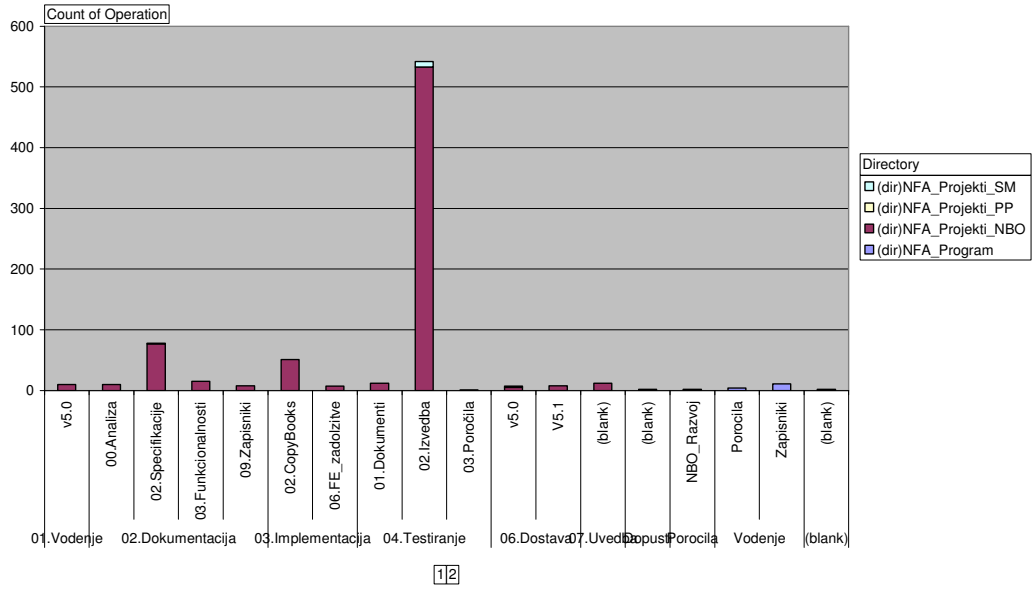
Nekaj primerov poročil za vodstvo [29]:



Slika 2.5: Prikaz aktivnosti po uporabniških imenih v enem tednu



Slika 2.6: Prikaz aktivnosti v enem tednu po uporabniku



Slika 2.7: Prikaz aktivnosti na različnih mapah modula

2.7 Izkušnje in subjektivna ocena uvajanja nove rešitve

Sistem je bil uveden v bančno okolje v približno pol leta od trenutka odločitve, da je izbran CVSNT. Po izobraževanju uporabnikov so ti povedali za nov sistem tudi članom na drugih projektih in približno leto kasneje so se tudi trije drugi projekti odločili, da bodo svojo dokumentacijo vodili v sistemu za nadzor različic. Bil sem presenečen, da tako velika in geografsko razpršena organizacija nima enotnega sistema za spremljanje in upravljanje nastanka dokumentacije predvsem na njihovih projektih, ki vključujejo razvoj s podizvajalci. Najbolj občutljivi del posla je pridobitev vseh znanj, ki jih mora prodajalec

predati kupcu neke storitve. Neprevidnost pri vpogledu nastanka dokumentacije in nejasen cilj obvladovanja tega, kar smo kupili, nas lahko na dolgi rok stane veliko denarja.

Sama tehnična argumentiranja so bila na zelo kvalitetni ravni in priznati moram, da sem se na podjetju NLB d.d. naučil počakati, da neko idejo najprej do konca raziščem in se je lotim kasneje, ko sem že preučil večino najbolj očitnih težav. IT strokovnjaki dobro skrbijo za celoten sistem in ga obvladujejo. V vsakem primeru previdnost pri vpeljavi tako razpredenega sistema ni nikoli odveč. Presenečen sem bil le nad tem, da mi nikoli ni bilo potrebno zagovarjati potrošnje prostora skladišč. Kljub temu, da skladišče zasede bistveno manj prostora kot klasičen način shranjevanja celotne vsebine vsake revizije, se skladišča, ko jih ljudje začnejo uporabljati, precej hitro širijo.

Glede samih uslužbencev podjetja in njihovega dojemanja potrebe po dodatnem sistemu v okolju imam občutek, da je uporabnikom najpogosteje všeč možnost, da vidijo kdo in kdaj je neko datoteko spreminjal. Začetna skrb je seveda ta, da nekdo želi normirati njihovo delo in jih skrivoma nadzorovati. Ta skrb prav kmalu izgine, saj je v novem sistemu uporabniku na voljo več orodij za zagovarjanje, kje in na čem je potrošil svoj službeni čas. Vsi projekti so vedno v zamudi in žal je pritisk na vse člane vedno prisoten. Vedno bo obstajala tudi skupina uporabnikov, ki jim spremembe niso všeč in ti bodo tudi neargumentirano trdili, da jim nov sistem vzame več časa in da je nepotreben.

Vodstvo je bilo z novim sistemom zadovoljno, poročanje o aktivnostih na skladišču je bilo nekaterim všeč in so si z njim pomagali do ocene nekega preteklega obdobja, spet drugim pa ne. Sami vodje navadno niso uporabljali funkcionalnosti, ki jim jo orodje ponuja. Le po potrebi so poklicali in prosili za točno določeno preverbo ali kontrolo neke datoteke, oziroma aktivnosti uporabnika v časovnem obdobju. V velikih sistemih je tako, da si vodje ne morejo vzeti dovolj časa, da bi dovolj dobro spoznali nova orodja in možnosti ter vzvode, ki jim jih nudijo. Osebno menim, da je to velika škoda, saj so navadno najbolj zanimiva tista spoznanja, ko nekdo sam »obrača podatke« in naleti na nek čuden vzorec v njih. Taka analiza sistema ne more biti izvajanja samodejno, ali od ljudi, ki ne vedo natančno, kakšno je stanje na projektu v poslovnem smislu.

Napovedujem, da se bo sistem nadalje širil po NLB d.d. in da bo čez nekaj časa najverjetneje zamenjan in nadgrajen s sistemom EVS. Umestitev v ostalo bančno okolje je dobro opisana v dokumentaciji in mojemu nasledniku ne bo predstavljala velikih težav. Ugibam lahko tudi, da je možnost, da gre sistem v opuščanje vedno prisotna. Nekdo mora redno bdeti nad sistemom in skrbeti, da se uporabniki držijo pravil. V trenutku, ko uporabniki ne bodo mogli izvajati opravil, ki jih ocenjujejo za kritične in bodo začeli iskati rešitev izven obstoječega sistema, bo sistem obsojen na izključitev iz okolja. Dober vzdrževalec in svetovalec je torej ključnega pomena, da bi se sistem lahko ohranil.

3 SKLEPNE UGOTOVITVE

V diplomski nalogi sem si zastavil cilj analizirati in prenoviti sistem dokumentacije podjetja NLB d.d.. Izvedba zastavljenih ciljev se je od samega začetka zdela precej preprosta. Analiza trenutnega stanja ni bila težaven del naloge, saj mi je vsakdo rad razložil, kaj počne in kako si je organiziral delo. Prenova sistema je zahtevala veliko več napora. Že med predlaganjem rešitev je problem postal bistveno bolj obširen. Sama namestitev in nastavitev sistema za nadzor različic je preprosta, bolj težavno je natančno argumentiranje vsake izbrane rešitve nadrejenim , testiranje različnih možnosti in predvsem zelo poglobljeno dokumentiranje vsakega področja celotnega sistema. O primernosti tega, kar želimo spremeniti je potrebno prepričati veliko število ljudi. V naravi zaposlenih je tudi to, da pretežno zavračajo vsako spremembo v sistemu, ki ga poznajo. Vsaka sprememba zahteva napor in prilagoditev. Vsakdo različno dojema, kaj je intuitivno, lažje, boljše, uporabnejše. Potrebno je imeti trdne živce in obširno znanje, da bi lahko različnim profilom uporabnikov suvereno predstavili, zakaj sploh morajo zamenjati svoj način shranjevanja in ostalega dela z dokumentacijo.

Ugotavljam, da me je z vidika sistematičnosti in pozornosti izkušnja ob vsakem delu zelo obogatila. Sedaj poskušam bolj natančno opisovati postopke, ki jih opravljam. Dobro raziskovanje vsake možne alternative prav tako prinaša širino v poznavanju problematike, kar olajša vsak pogovor z drugimi, različno mislečimi, strokovnjaki.

Kot možnost nadaljevanja dela, ki je tukaj začeto, vidim predvsem možnost integracije s sistemom za sledenje napak. Datoteke in priponke v takem sistemu bi se lahko avtomatično posodabljale tudi v sistemu za nadzor različic in obratno. Potrebno bi bilo premisliti, ali je taka implementacija potrebna in koristna.

Prav tako bi tudi dodatek v aplikacijah paketa Microsoft Office prišel izjemno prav pri takem sistemu za nadzor različic, kot je postavljen sedaj. Tak dodatek bi ob vsakem shranjevanju ponudil možnost objave datoteke in zapis komentarja. Integriranje pregleda zgodovine datoteke naravnost iz vmesnika aplikacij Excel, Word in ostalih aplikacij paketa Microsoft Office, bi ravno tako znal biti zanimiv doprinos k lažji in bolj samoumevni uporabi novega sistema. Tak sistem bi pri uporabniku deloval še bolj celovito kot sedaj.

Viri:

[1] Maja Pivec, Vladislav Rajkovič; Obvladovanje znanja z metodami umetne inteligence
[2] (2009) http://better-scm.berlios.de/comparison/comparison.html Shlomi Fish
[3] (2009) http://en.wikipedia.org/wiki/Comparison_of_revision_control_software
[4] (2009) http://en.wikipedia.org/wiki/Systems_Management_Server
[5] (2009) http://en.wikipedia.org/wiki/Scriptlogic
[6] (2009) http://better-scm.berlios.de/comparison/comparison.html
[7] (2009) http://en.wikipedia.org/wiki/Version_control_system
[8] (2009) http://msdn.microsoft.com/en-us/library/aa380493.aspx
[9] Interni vir NLB d.d.: Aleksander Pahor (2006), Navodila uporabe orodja TortoiseCVS
[10] Interni vir NLB d.d.: Aleksander Pahor (2007), Administracija sistema za nadzor različic CVSNT
[11] March Hare Software Ltd. - Več avtorjev, (2005), cvsnt–Concurrent Versions System 2.5.03.2151
[12] Moshe Bar, Karl Fogel, (2003), Open Source Development with CVS – 3rd Edition
[13] Interni vir NLB d.d., Aleksander Pahor (2006); Namestitev strežnika CVSNT
[14] Graham Crockford, (2005); TortoiseCVS Administration
[15] Goran Bavčar (2007) – SQL Adria – prezentacija »Več kot SMF zapisi«
[16] (2009) http://web.telia.com/~u86216121/ViewCvsSetup.html
[17] (2009) http://web.telia.com/~u86216121/cvsmailer/CVSMailer.html
[18] David Russell, Nilesh Patel, “Increasing Software Engineering Efficiency Through Defect Tracking Integration," icsea,pp.5, International Conference on Software Engineering Advances (ICSEA'06), 2006
[19] Nadzor različic s programskim paketom CVS, Peterlin Primož, diplomska naloga, Monitor. - ISSN 1318-1017. - #Letn. #13, #št. #3 (mar. 2003), str. 116-119.
[20] Vpeljava sistema Subversion za upravljanje izvorne kode v razvoj programske opreme : diplomska naloga, Kraševič Tomaž, 2007, COBISS.SI-ID: 5825364
[21] Zasnova samodejne posodobitve datotek na oddaljenih sistemih : diplomska naloga univerzitetnega študijskega programa, Čapelnik Goran, 2005, COBISS.SI-ID: 9638934
[22] ELECTRONIC DATA PROCESSING IN BANKS By Ralph C. Boyer; http://www.smecc.org/new_page_14.htm
[23] Rapid Subversion Adoption Validates Enterprise Readiness and Challenges Traditional Software Configuration Management Leaders, EETimes (2007-05-17).
[24] Tamara Železnik Kohek: magistrsko delo (2006); Operativna tveganja v Novi Ljubljanski Banki, povezana s prevzemom evra
[25] Atria Software (1994); CASEVision / ClearCase Administration Guide
[26] Interni vir Hermes Softlab d.d.: HSL-HRDB (Hermes Softlab - Human Resources DataBase) – kadrovska baza
[27] (2009) http://www.march-hare.com/cvspro/svn.htm
[28] (2009) http://www.thefreedictionary.com/documentation alineja 3.
[29] Interni vir NLB d.d.: Aleksander Pahor (2007), Poročilo projektnemu vodstvu NBO

Slike:

Tabele:

Priloga 1: