UNIVERZA V LJUBLJANI
EKONOMSKA FAKULTETA

MAGISTRSKO DELO

Odločitveni model za izbiro sistema orodij za upravljanje s konfiguracijo izdelkov

Ljubljana, januar 2003 Uroš Prelovšek

Izjava

Študent Uroš Prelovšek izjavljam, da sem avtor tega magistrskega dela, ki sem ga napisal pod mentorstvom prof. dr. Vladislava Rajkoviča in skladno s 1. odstavkom 21. člena Zakona o avtorskih in sorodnih pravicah dovolim objavo magistrskega dela na fakultetnih spletnih straneh.

V Ljubljani, dne 14.1.2003 Podpis: _____

Povzetek

Magistrsko delo obravnava problematiko izbire sistema orodij za upravljanje konfiguracij izdelkov SI2000 v podjetju Iskratel, d.o.o., Kranj. Pomembni značilnosti tovrstnih odločitvenih problemov, ki so običajno del širše zastavljenih projektov uvajanja sistemov za podporo poslovanja s proizvodi, sta: obsežnost in kompleksnost problematike, ki zahteva interdisciplinaren pristop, ter hitre spremembe, tako v tehnologiji obravnavanih sistemov, kot glede pogojev v zvezi z nakupom in vpeljavo v poslovni proces. Pri projektih uvajanja teh sistemov v uporabo v prvi vrsti ne gre le za spremembo informacijske tehnologije, temveč predvsem za kulturno, sociološko spremembo v podjetju, prav tako pa za spremembo poslovnih procesov. Z namenom čim bolj sistematičnega, celovitega in transparentnega ocenjevanja nastopajočih različic je bila izbrana metoda večparametrskega odločanja s pomočjo ekspertnega sistema DEX. Postavljen je bil odločitveni model, s pomočjo katerega je bilo ocenjenih pet podpornih sistemov, t.j. pet variant, ki so se z vidika uporabe pokazale kot najbolj zanimive. Za razvrščanje ocenjenih variant je bila izbrana metoda portfelja, pri čemer sta bila upoštevana ocena konkurenčne prednosti posameznega podpornega sistema in ocena ustrezanja zahtevam poslovnega procesa in sistema v podjetju Iskratel. Magistrsko delo oziroma postavljeni odločitveni model omogoča nov, razširjen in poglobljen pogled na obravnavano problematiko. Uporabljena metodologija in orodje DEX sta se na navedenem primeru pokazala kot zelo primerna za uporabo. Njuna uporaba bi pripomogla k še bolj sistematičnemu in transparentnejšemu načinu odločanja v nadaljnih projektih prenove ali vpeljave sistemov za podporo poslovanju v omenjenem telekomunikacijskem podjetju.

Abstract. Decision model for selection of product configuration management system

This master thesis deals with the problem of evaluation and selection of configuration management system, for the family of products SI2000 in Iskratel company. Two of the important characteristics of such decision problems, which usually constitute an integral part of larger projects of introducing business support systems into the business systems, are: extensiveness and complexity of problems, which require the use of interdisciplinary proceedings, and also fast changes in purchase conditions and technology of systems. Introduction of such business support systems into business process does not only mean a change in information technology, but more importantly, a change in business system culture and business processes. To ensure systematic work, the integrity and transparency of configuration management systems' evaluation and selection process, the methodology of multi-attribute decision making with expert system DEX has been chosen. A decision model has been built, by which five most interesting variants have been evaluated. To classify the evaluated variants, the portfolio method has been used. The most important factors, considered in the portfolio, have been the competitive position of the configuration management system, and the extent to which they meet the requirements of the business system and process in Iskratel company. The thesis and the decision model enable a new, broader and deeper insight into discussed problems. The methodology and the tool, used for evaluation and selection of configuration management systems, have been recognized as very suitable for this decision making case. Their usage would contribute to even more systematic and transparent decision making within the subsequent projects of modernization or introduction of new business support systems into the abovementioned telecommunications company.

Vsebina

1. Uvod ... 1

1.1 Struktura magistrskega dela ... 3

1.2 Kratka predstavitev poslovnega in tehnološkega okolja in strukture magistrskega dela ... 3

1.2.1 Podjetje in njegovi izdelki ... 3

1.2.2 Tehnologija orodij, ki se uporabljajo v procesu načrtovanja, razvoja in vzdrževanja telekomunikacijskih izdelkov ... 4

1.2.3 Področje upravljanja konfiguracij telekomunikacijskih izdelkov ... 4

1.3 Zakaj je problem težak? ... 4

1.4 Odločitvena skupina ... 6

1.4.1 Lastnik odločitvenega problema, odločitvena skupina in uporabniki, ki jih odločitev neposredno zadeva ... 6

1.4.2 Sodelavci v odločitveni skupini ... 6

1.5 Metoda dela ... 6

1.5.1 Elementi uporabljene metodologije ... 7

1.5.1.1 Kaj je odločitveni problem ... 7

1.5.1.2 Večparametrsko odločanje ... 7

1.5.1.3 Pristop k večparametrskemu odločanju s sistemom DEX ... 7

1.5.2 Ključni koraki pri izvajanju odločitev s pomočjo sistema DEX ... 8

1.5.2.1 Identificiranje problema ... 8

1.5.2.2 Identificiranje in strukturiranje kriterijev ... 8

1.5.2.3 Določitev funkcij koristnosti ... 9

1.5.2.4 Opis variant ... 9

1.5.2.5 Vrednotenje in analiza variant ... 9

2. Področje upravljanja s konfiguracijo izdelkov ... 10

2.1 Pomen upravljanja s konfiguracijo izdelkov ... 10

2.1.1 Kaj je upravljanje s konfiguracijami in čemu je namenjeno ... 10

2.1.2 Elementi upravljanja konfiguracij in pomen planiranja ... 12

2.1.3 Proces upravljanja konfiguracij in poslovno okolje ... 13

2.1.3.1 Vloga procesa upravljanja konfiguracij v poslovnem okolju ... 13

2.1.3.2 Povezanost upravljanja konfiguracij z drugimi procesnimi področji ... 14

2.1.4 Upravljanje konfiguracij za programsko opremo ... 15

2.1.4.1 Posebnosti upravljanja konfiguracij programske opreme ... 16

2.1.4.2 Dvojna vloga upravljanja konfiguracij programske opreme ... 17

2.1.5 Upravljanje družin proizvodov, ki vsebujejo vložene računalniške sisteme, in njihovih konfiguracij ... 17

2.1.5.1 Splošne značilnosti ... 17

2.1.5.2 Razvojni proces ... 20

2.1.5.3 Vzdrževanje ... 20

2.1.6 Sistemi za upravljanje konfiguracij izdelkov in njihove programske opreme ... 22

2.1.6.1 Sistemi za upravljanje podatkov o proizvodih (PDM sistemi) ... 23

2.1.6.2 Sistemi za upravljanje konfiguracij programske opreme (SCM sistemi) ... 25

2.2 Kratek pregled pristopov k upravljanju s konfiguracijo izdelkov in njihove programske opreme ... 26

2.2.1 Feilerjeva teorija modelov upravljanja konfiguracij ... 27

2.2.1.1 Model odjava/prijava ... 27

2.2.1.2 Kompozicijski model ... 29

2.2.1.3 Transakcijski model ... 31

2.2.1.4 Model množic sprememb ... 34

2.2.1.5 Uporabnost naštetih pristopov ... 36

2.2.2 Taramova razvrstitev pristopov k upravljanju konfiguracij vloženih računalniških sistemov ... 36

2.2.2.1 Nadzor verzij ... 37

2.2.2.2 V izdajanje usmerjeno upravljanje konfiguracij programske opreme ... 37

2.2.2.3 V spremembe usmerjeno upravljanje konfiguracij programske opreme ... 38

2.2.2.4 Upravljanje podatkov o proizvodih (Product Data Management) ... 38

2.2.2.5 Globalno upravljanje proizvodov ... 39

3. Analiza obstoječega stanja v podjetju 39
3.1 Funkcije obstoječega sistema in značilnosti procesa 39
3.1.1 Izdelki SI2000 in njihov življenjski cikel 39
3.1.2 Sistemi za podporo upravljanju z življenjskim ciklom izdelkov in njihovimi konfiguracijami 40
3.1.2.1 Splošne lastnosti 40
3.1.2.2 PDM sistem SOPRAN 42
3.1.2.3 SCM sistem 42
3.1.2.4 Povezave med PDM in SCM sistemom 43
3.1.3 Ključne značilnosti procesa upravljanja konfiguracij in sprememb izdelkov in njihove PO 44
3.1.3.1 Ključne vloge v procesu 44
3.1.3.2 Proces upravljanja inženirskih sprememb na izdelkih 44
3.1.3.3 Značilnosti metodologije in procesa upravljanja konfiguracij PO 46
3.2 Kritična analiza obstoječega stanja 47
3.2.1 Razvojni proces in upravljanje konfiguracij izdelkov 47
3.2.2 Trenutni podporni sistem 49
3.2.2.1 Splošne ugotovitve 49
3.2.2.2 PDM Sopran 49
3.2.2.3 SCM sistem 49
3.2.2.4 Povezave med sistemoma in perspektiva sistema 50
3.2.3 Stanje ostalih procesov v neposredni povezavi z upravljanjem konfiguracij izdelkov 50
3.2.4 Identificirani predlogi izboljšav 51
4. Izgradnja odločitvenega modela 52
4.1 Kriteriji 52
4.1.1 Uporabljeni kriteriji 52
4.1.1.1 Kriteriji v zvezi s konkurenčno sposobnostjo 52
4.1.1.2 Kriteriji v zvezi z izpolnjevanjem zahtev podjetja 53
4.1.2 Kratek opis kriterijev 53
4.1.2.1 Kriteriji v zvezi s konkurenčno sposobnostjo 53
4.1.2.2 Kriteriji za izpolnjevanje zahtev podjetja 56
4.1.3 Ureditev kriterijev 60
4.2 Funkcije koristnosti 61
4.2.1 Glavne značilnosti funkcij koristnosti 61
4.2.1.1 Značilnosti funkcij koristnosti za dejavnike konkurenčne sposobnosti 61
4.2.1.2 Značilnosti funkcij koristnosti v zvezi z zahtevami podjetja 62
4.2.2 Primerjava pomembnosti kriterijev 63
5. Uporaba odločitvenega modela 64
5.1 Izbira variant 65
5.2 Sistemi za podporo upravljanja konfiguracij izdelkov in njihove programske opreme 66
5.2.1 Orodje SCCS 67
5.2.1.1 Osnovne funkcije in mehanizmi orodja SCCS 67
5.2.2 Sherpa PIMS 68
5.2.2.1 Značilnosti zgradbe sistema 68
5.2.2.2 Predstavitev osnovnih funkcij sistema 68
5.2.3 Rational ClearCase 70
5.2.3.1 Rational UCM 70
5.2.3.2 Zgradba in funkcionalnosti orodja Rational ClearCase 73
5.2.4 EDS Teamcenter 75
5.2.4.1 Namen sistema in osnovni koncepti 75
5.2.4.2 Zgradba sistema 76
5.2.4.3 Funkcionalnosti sistema 77
5.2.4.4 Proces sprememb v Teamcentru 79
5.2.5 mySAP PLM 81
5.2.5.1 Poslovni sistem SAP in njegov pristop k upravljanju življenjskega cikla produktov 81
5.2.5.2 Zgradba sistema, načini povezovanja in prenosa podatkov 81
5.2.5.3 Funkcije sistema 82
5.2.6 Telelogic CM Synergy 84
5.2.6.1 Predstavitev osnovnih konceptov, vgrajenih v CM Synergy 84
5.2.6.2 Zgradba CM Synergy 85
5.2.6.3 Funkcije sistema CM Synergy 86

5.3 Predstavitev variant *87*
5.3.1 PDM Sherpa in lastni SCM - nadaljevanje razvoja z izvedbo nadgradnje 88
5.3.2 PDM Teamcenter in SCM ClearCase 88
5.3.3 PDM mySAP PLM in CM Synergy 88
5.3.4 PDM mySAP PLM in SCM ClearCase 89
5.3.5 PDM Teamcenter in lastni SCM 89
5.4 Rezultati vrednotenja *89*
5.4.1 Ocene variant 89
5.4.2 Najboljša varianta; primerjava s preostalimi variantami 91
6. Kritična analiza modela, metodologije in narave odločanja 93
6.1 Analiza modela in odločitve *93*
6.1.1 Ustreznost modela in obrazložitev odločitve 93
6.1.2 Prednosti in pomanjkljivosti variant 94
6.1.3 Bistvene razlike med variantami 95
6.1.4 Občutljivost odločitve 97
6.2 Izkušnje z metodologijo DEX in uporabo orodja DEXi *98*
6.3 Analiza narave odločanja pri izbiri sistemov za podporo upravljanju konfiguracij izdelkov *99*
7. Zaključek in predlogi za nadaljnje delo 100
Literatura 102
Viri 105
Priloga A - Odločitveni model 1
Priloga B - Rezultati vrednotenja 12
Priloga C - Razlaga pojmov, kratic 17
Priloga D - Kratka predstavitev razvoja in uporabe upravljanja konfiguracij izdelkov 24
i. Disciplina *24*
ii. Podporni sistemi *25*
Priloga E - Procesni pristop RUP (Rational Unified Process) 27
Priloga F - Rational ClearQuest 31
Priloga G - Najboljše izkušnje ali pravila dobre prakse na področju upravljanja konfiguracij izdelkov 33
Priloga H - Projekti uvajanja sistemov za upravljanje konfiguracij izdelkov 37
i. Pričakovanja organizacij oziroma zakaj podjetja vlagajo v sisteme za upravljanje konfiguracij izdelkov *37*
ii. Ključni dejavniki v projektih uvajanja CM sistemov *39*

1. Uvod

Namen magistrskega dela je na sistematičen način ovrednotiti in izbrati najustreznejši sistem orodij za upravljanje konfiguracij Iskratelovih telekomunikacijskih izdelkov SI2000. V praksi se je pokazalo, da obstoječi tovrstni podporni sistem ne ustreza več vse ostrejšim poslovnim in tehnološkim zahtevam. To je v hitro se spreminjajočem poslovnem svetu povsem normalno, zahteva pa pravočasno zaznavo novih zahtev za sistem in ustrezno ukrepanje.
Hkrati s tem želimo dobro preučiti kontekst, v katerem delujejo sistemi za podporo upravljanju konfiguracij in še bolje spoznati tako disciplino kot proces upravljanja konfiguracij ter v sklopu tega identificirati povezave z drugimi poslovnimi procesi, tako nasploh kot v našem podjetju.
Cilji, ki jih želimo doseči z odločitvijo, so:

- v dveh letih dobiti povezan, celovit in učinkovit sistem za upravljanje konfiguracij izdelkov SI2000, tako za programsko kot za strojno opremo in dokumentacijo;
- preučiti identificirane probleme podjetja, ki so del procesa upravljanja konfiguracij ali mejijo nanj;
- spoznati značilnosti procesa odločanja in najpomembnejših dejavnikov, ki nanj vplivajo;
- dobiti pregled nad najnovejšimi pristopi, ki jih snujejo in uporabljajo podjetja, ki razvijajo najboljša orodja za upravljanje konfiguracij izdelkov.

Hkrati z vpeljavo podpornega sistema želimo postaviti nov, celovit in usklajen koncept na procesnem področju upravljanja konfiguracij in ga realizirati tako v procesu kot v orodjih. Pri tem mora biti naložba v sistem za podjetje ekonomsko upravičena. Stroški v podobne podporne sisteme vplivajo na stroške razvoja in vzdrževanja izdelkov, kar vpliva na ceno samih proizvodov in s tem konkurenčno sposobnost izdelkov na trgu.
V praksi se za upravljanje konfuguracij uporablja več vrst podpornih sistemov (v nadaljevanju imenovanih tudi *CM sistemi*), npr. sistemi za upravljanje podatkov o proizvodih (PDM), sistemi za upravljanje programske opreme (SCM), proizvodni informacijski sistemi (ERP) ipd. Zato je potrebno določiti okvir sistemov, ki jih bomo obravnavali v nadaljevanju. Zahteve za obravnavane podporne sisteme lahko označimo z naslednjimi njihovimi funkcijami in vmesniki:

- upravljanje konfiguracij celotnega izdelka: upravljanje s kosovnicami celotnega proizvoda, kosovnicami programske in strojne opreme (krat. PO, SO) na najvišjem nivoju, pri čemer so pod strojno opremo mišljeni tako elektronski (integrirana vezja) kot mehanski deli (npr. ohišja izdelkov);
- upravljanje konfiguracij programske opreme izdelka, vključno s sistemi za nadzor verzij PO, sistemi za gradnjo in izdajanje programskih paketov ipd.;
- nadzor verzij gradnikov in kosovnic strojne in programske opreme izdelkov SI2000, vključno s produktno, razvojno, metodološko in uporabniško dokumentacijo;
- povezano, celovito upravljanje sprememb programske opreme in celotnega izdelka;
- ustrezna raven podpore vzporednemu načinu dela razvijalcev;
- možnost upravljanja z zahtevami za izdelke;
- upravljanje z dokumentacijo, pri čemer je najpomembneje zagotoviti sledljivost, tako v zvezi z izdelavo kot v zvezi z njeno veljavnostjo;
- vmesniki sistema: vmesniki proti okoljem za razvoj programske in strojne opreme; vmesnik proti orodju za prijavo napak v sistemu (angl. helpdesk); vmesnik proti sistemu za upravljanje projektov; vmesniki proti sistemom, ki se uporabljajo za nadzor

dobaviteljev; vmesniki proti ključnim sistemom, ki jih uporabljajo neposredni notranji odjemalci razvojnih storitev (npr. proizvodnja).

Omenjene funkcije bi morale delovati učinkovito v kontekstu načrtovanja in razvoja *družine telekomunikacijskih izdelkov*. Podporni sistem orodij mora ustrezati tudi splošnim zahtevam, ki se pričakujejo od vsakega tovrstnega sistema, kot npr.: zanesljivost delovanja orodij; zagotavljanje konsistentnosti podatkov o izdelkih; varnost hranjenja in dostopanja do informacij; mehanizme za promocijo oziroma upravljanje življenjskega cikla posameznih gradnikov konfiguracije ipd. Pri vsem tem uporabniki obstoječega podpornega sistema razumljivo pričakujejo, naj bi obdržali vse bistvene dobre lastnosti sedanjega podpornega sistema.

V smislu zgoraj navedenega namena in ciljev magistrskega dela so bile v preteklem letu izvedene naslednje aktivnosti:

- identificiran je bil proces upravljanja konfiguracij izdelkov v podjetju. Pri tem so bili izvedeni intervjuji s širokim krogom uporabnikov obstoječega CM sistema;
- tekoče so se spremljale informacije z obravnavanega strokovnega področja;
- opravljene so bile analize načinov dela z razvojnimi orodji, ki se uporabljajo v razvojnem procesu podjetja (Microsoft Developer Studio, Tornado, Java, C/C++ ipd.);
- ustanovljena je bila odločitvena skupina v zvezi z izbiro CM sistemov, v kateri tudi sam aktivno sodelujem. Njen namen in delo sta podrobneje predstavljena v nadaljevanju;
- s pomočjo metodologije DEX je bil postavljen večparametrski odločitveni model za izbiro sistema za upravljanje konfiguracij izdelkov. S tem modelom je bilo izvedeno ovrednotenje petih različic tovrstnih sistemov, ki so se za uporabo pokazale kot najprimernejše.

Pri delu so mi znatno pomagali znanje in izkušnje, ki sem jih na omenjenem področju pridobil v preteklih letih, predvsem z izvedbo ali sodelovanjem v naslednjih aktivnostih in usposabljanjih:

- izvedbe analiz obstoječega procesa in podpornega sistema za upravljanje konfiguracij izdelkov SI2000;
- vrednotenje nekaterih za podjetje najbolj zanimivih komercialnih sistemov orodij za upravljanje konfiguracij izdelkov ali programske opreme (ClearCase, CM Synergy, PVCS itd.), vključno z ogledom nekaterih orodij v drugih podjetjih (npr. MKS Source Integrity v Hermes Softlabu) in ogledom predstavitev zastopnikov orodij za upravljanje konfiguracij v našem podjetju (predstavniki podjetij Rationala in Maranda za sistem ClearCase, podjetja Telelogic za CM Synergy, TIS za PVCS, EDS za Teamcenter itd).
- sodelovanje na nekaterih konferencah na temo upravljanja konfiguracij izdelkov in njihove PO (Pariz '97, Toulouse '99, Minneapolis 2000), v okviru katerih so bili predstavljeni tudi različni komercialni sistemi za upravljanje konfiguracij;
- seznanjanje z zahtevami standardov in modelov na področju upravljanja konfiguracij izdelkov (IEEE standardi, CMM model, standard ISO 9000 ipd.);
- šolanje v zvezi z metodologijo identificiranja poslovnih procesov (SIQ);
- postavitev, administriranje in vzdrževanje sistema za upravljanje konfiguracij z uporabo orodja ClearCase, za projekt razvoja tehnologije ADSL v Siemensu (Muenchen) - od avgusta 1999 do konca februarja 2000.

V nadaljevanju uvodnega poglavja je v kratkem opisana struktura magistrskega dela in orisan poslovno-tehnološki kontekst, v katerem se izvaja odločitveni proces. Podani so tudi najpomembnejši vzroki za težavnost obravnavane problematike. Na koncu poglavja sta predstavljeni odločitvena skupina in uporabljena metodologija.

1.1 Struktura magistrskega dela

V nadaljevanju je v kratkem orisana vsebina poglavij magistrskega dela:

- v Uvodu je podan namen in cilj magistrskega dela. Opredeljena je obravnavana problematika, vključno s svojim poslovno-tehnološkim kontekstom. Pojasnjeno je, zakaj je problem težak. Predstavljeni sta odločitvena skupina in metoda dela;
- v drugem poglavju je predstavljeno področje upravljanja s konfiguracijo industrijskih izdelkov in njihove običajno zelo pomembne komponente, programske opreme. Zajet je tako procesni vidik kot vidik podpornih sistemov, ki ta proces avtomatizirajo. Orisana je Feilerjeva teorija modelov upravljanja konfiguracij in Taramova razvrstitev pristopov in sistemov za upravljanje konfiguracij vloženih računalniških sistemov;
- v tretjem poglavju so opisane ključne značilnosti življenjskega cikla proizvodov SI2000, njihovega upravljanja in obstoječega procesa oziroma sistema upravljanja konfiguracij v omenjenem telekomunikacijskem podjetju, pri čemer je podana analiza tega stanja;
- v četrtem poglavju je podana zgradba odločitvenega modela, razvitega in uporabljenega za izbiro sistema orodij za upravljanje konfiguracij izdelkov. Opisani so kriteriji za ovrednotenje, njihova ureditev, značilnosti uporabljenih funkcij koristnosti in drugi elementi modela;
- v petem poglavju je opisana uporaba modela. Predstavljene in na podlagi omenjenega modela ovrednotene so najaktualnejše variante oziroma podporni sistemi;
- v šestem poglavju je podana analiza modela, njegove uporabe in dobljenih rezultatov;
- zadnje, sedmo poglavje je namenjeno zaključnim ugotovitvam in predlogom za nadaljnje delo.

1.2 Kratka predstavitev poslovnega in tehnološkega okolja in strukture magistrskega dela

1.2.1 Podjetje in njegovi izdelki

Podjetje Iskratel, d.o.o., Kranj, razvija, proizvaja, trži in vzdržuje telekomunikacijske komutacijske izdelke serije SI2000. Ti proizvodi se uporabljajo za prenos govora, vse bolj pa tudi podatkov, tako v slovenskih omrežjih kakor tudi v omrežjih mnogih tujih držav. V tehnološkem smislu gre za zapletene izdelke, ki so običajno sestavljeni iz večih komponent in tisočerih med seboj povezanih gradnikov (tudi: elementov ipd.), podvrženih neprenehnim spremembam.

Ko govorimo o razvoju telekomunikacijskih sistemov, je potrebno razlikovati med tehnologijo samih proizvodov in tehnologijo podpornih sistemov oziroma sistemov za podporo poslovanju, ki se uporabljajo pri upravljanju teh izdelkov skozi njihov življenjski cikel. Tehnologija telekomunikacijskih izdelkov v tem delu ne bo predmet obravnave. Predstavljeno je le okolje oziroma kontekst, v katerem ti proizvodi nastajajo, da bi lahko bolje razumeli problemsko stanje. Z določenih zornih kotov bo obdelano drugo od omenjenih področij, t.s. podporni sistemi, ki se uporabljajo pri upravljanju proizvodov SI2000. Pri tem se bomo osredotočili na sisteme, namenjene podpori procesu upravljanja konfiguracij izdelkov in njihove zelo pomembne komponente, programske opreme. Ti sistemi imajo nadvse pomembno vlogo pri obvladovanju tako samih proizvodov kot tudi procesov snovanja teh izdelkov, njihovega uvajanja na tržišče, pa tudi pri kasnejšem vzdrževanju, vse do umika iz uporabe.

1.2.2 Tehnologija orodij, ki se uporabljajo v procesu načrtovanja, razvoja in vzdrževanja telekomunikacijskih izdelkov

Tehnologija računalniških sistemov, aplikacij in okolij, ki se uporabljajo pri razvoju in vzdrževanju komutacijskih sistemov, je prepletena z vse hitrejšim razvojem informacijske tehnologije. Razvoj omenjenih aplikacij gre v smeri vse učinkovitejše podpore življenjskih ciklov komutacijskih sistemov in z njimi povezanih procesov. Za te poslovne procese postaja vse bolj značilno, da ne potekajo več le na eni, temveč na večih geografsko ločenih lokacijah. Tako npr. pri procesu razvoja izdelkov SI2000 sodelujejo mnogi manjši ali večji Iskratelovi poslovni partnerji, med njimi tudi mnoga hčerinska podjetja, tako doma kot v tujini. Za tovrstno okolje, pri katerem se presegajo krajevne, kulturne in druge meje, in je za sodobni poslovni svet vse bolj značilno, se uporabljajo različna poimenovanja, kot npr. *geografsko porazdeljen (distribuiran) razvoj, virtualni razvoj, virtualno sodelovanje* (McMahon, 2001, str. 4).

1.2.3 Področje upravljanja konfiguracij telekomunikacijskih izdelkov

Upravljanje konfiguracij izdelkov je pomemben vidik upravljanja zapletenih proizvodov s področja vojaške, letalske, avtomobilske, telekomunikacijske in drugih vrst industrij. Za te izdelke je značilno, da v vse večji meri vsebujejo vložene računalniške sisteme. Eno od njihovih najpomembnejših komponent predstavlja t.im. vložena PO. Področje *upravljanja konfiguracije PO* se lahko obravnava kot sestavni del upravljanja konfiguracij proizvodov. Ker vložena PO običajno predstavlja tudi najkompleksnejši del tovrstnih izdelkov, lahko ugotovimo, da upravljanje s konfiguracijami PO izdelkov predstavlja enega kritičnih elementov upravljanja konfiguracij izdelkov.

Upravljanje konfiguracij klasičnih industrijskih izdelkov, še posebej na področju vojaške in letalske industrije, je starejša disciplina kot upravljanje konfiguracij PO izdelkov. Slednja se je bolj razvila in dozorela šele v zadnjih dvajsetih letih, hkrati z večanjem pomena PO. Obe področji upravljanja sta podrobneje predstavljeni v drugem poglavju.

V praksi je upravljanje konfiguracij podprto z avtomatiziranimi računalniškimi sistemi, ki povečujejo učinkovitost tega vidika upravljanja z izdelki, poleg tega pa lahko vsebujejo tudi nekatere druge funkcije. Na ravni konfiguracij proizvodov (pri čemer je poudarek na upravljanju sprememb izdelkov, obvladovanju konfiguracij vgrajenih komponent, nadzoru dokumentacije ipd.) se pogosto uporabljajo PDM, PLM ali sorodni tipi sistemov. Na nivoju konfiguracij PO pa se uporabljajo t.im. SCM sistemi.

1.3 Zakaj je problem težak?

Problem je zelo zapleten predvsem zaradi naslednjih dejavnikov:

- proces upravljanja konfiguracij izdelkov in njihove PO je tesno vpet v procese razvoja in vzdrževanja teh izdelkov, zato je omenjene procese skorajda nemogoče obravnavati ločeno. To se odraža tudi na podpornih in razvojnih sistemih. Razvojna okolja imajo pogosto v sebi vključene funkcije za upravljanja konfiguracij izdelkov ali njihovih komponent, sodobnejši sistemi za podporo upravljanja konfiguracij izdelkov pa imajo v sebi vključene funkcije za avtomatizacijo razvojnega procesa;
- naše zahteve za sistem so tako zapletene, da na tržišču sistemov ne obstaja en tip podpornih sistemov, ki bi v ustrezni meri ustrezal tem zahtevam. Zato je potrebno obravnavati različice, sestavljene iz večih modulov, izdelanih s strani različnih proizvajalcev;

- način izvajanja razvojnega procesa v podjetju, prav tako pa tudi z njim močno povezanega procesa upravljanja konfiguracij izdelkov, je znan le v obrisih in ni povsod strogo določen;
- brez praktičnega preizkusa je nemogoče dovolj zanesljivo oceniti vse prednosti in slabosti drugih sistemov v primerjavi s sistemom, ki se v ta namen trenutno uporablja. Način razvoja družine izdelkov v podjetju onemogoča poskusno vpeljavo podpornih sistemov preko posameznega projekta. Večina ključnih projektov, povezanih z izdelki SI2000, je med seboj tesno povezanih, kar je posledica sorodnosti izdelkov, tako po sestavi kot po funkcijah. Preostali projekti se od ključnih bistveno razlikujejo, tako po obsegu in kompleksnosti kot po metodologiji dela. To pomeni, da rezultati pilotskega projekta, izvedenega na teh projektih, ne bi dali zanesljive ocene za projekte in proizvode SI2000. Praktičen preizkus podpornih sistemov je torej možno narediti le z razmeroma velikimi stroški za podjetje - že minimalni, enomesečni preizkus podpornih sistemov bi vplival na opazno upočasnitev ključnih projektov;
- CM sistem v veliki meri uporabljajo tako izvajalci nalog (razvijalci in vzdrževalci izdelkov) kot vodje (vodje funkcijskih enot, produktni vodje, vodje razvojnih projektov...), oba tipa uporabnikov pa je hkrati težko povsem zadovoljiti. Razvijalci od CM sistema pričakujejo učinkovitost pri izvajanju funkcij v inženirskem procesu, dobro integracijo z obstoječimi razvojnimi okolji, enostavno ravnanje in prijaznost do uporabnika, vodje produktov in projektov pa od orodij v prvi vrsti pričakujejo učinkovit nadzor nad procesom, pregledna poročila s ključnimi informacijami o projektu in proizvodu ipd.;
- v Sloveniji po velikosti in zapletenosti projektov ni primerljivih podjetij na področju razvoja telekomunikacijskih izdelkov ali izdelkov z vloženo PO. Zato se na izkušnje drugih podjetij lahko opremo le v zelo omejeni meri;
- sami CM sistemi se hitro razvijajo, nove izdaje si pogosto sledijo. S tem je povezan problem, ali lahko za dani sistem dobimo popolnoma ažurne informacije. V zadnjih letih je bilo v večino sistemov vgrajenih npr. veliko funkcionalnosti za delo v Web okolju. Za podjetja, ki tovrstne sisteme proizvajajo, je značilna velika intenzivnost pri poslovnem povezovanju in tudi prevzemih. Primeri iz zadnjih let: podjetje Telelogic je prevzelo Continuus, EDS je priključil SDRC, ta pred tem Sherpo itd.;
- ocene sistemov orodij za upravljanje konfiguracij se v strokovni literaturi ali s strani svetovalnih hiš (CIMdata, Yphise, OVUM ipd.) zaradi zahtevnosti in obširnosti problematike pojavljajo zelo redko. Te so v mnogih primerih ali sponzorirane s strani proizvajalcev tovrstnih podpornih sistemov ali izdelane s strani tistih svetovalnih hiš, ki so v poslovnih partnerstvih ali celo v lasti katerega teh proizvajalcev. Zato je potrebno pri njihovi uporabi zavzeti kritično distanco. Poleg tega se pri podpornih sistemih navadno ocenjuje splošna kakovost funkcionalnosti, ne glede na to, kakšna je njihova sposobnost izpolnjevanja zahtev določenega tipa poslovnega okolja;
- informacije poslovne in finančne narave, ki odločilno vplivajo na nakup teh sistemov, so velikokrat tajne oziroma znane le najožjemu krogu pogajalcev. Podvržene so pogostim spremembam. Neposredno se dotikajo tako pogojev nakupa, vključno s ceno nakupa in vključenimi rešitvami, kot tudi pogojev in cene vzdrževanja. Pogoji prodaje se pogosto spreminjajo in so odvisni od večih dejavnikov, npr. od poslovnih interesov prodajalca v določenem delu sveta ali na določenem poslovnem področju, od kakovosti poslovnih odnosov med prodajalcem in kupcem ipd.

1.4 Odločitvena skupina

1.4.1 Lastnik odločitvenega problema, odločitvena skupina in uporabniki, ki jih odločitev neposredno zadeva

Končna odločitev o sistemu za upravljanje konfiguracij izdelkov SI2000 bo sprejeta s strani vodstva poslovne enote. Pred tem bodo izvedene analize in ovrednotenja sistemov, s tehničnega in poslovnega zornega kota. Za izvedbo le teh je zadolžena odločitvena skupina, ki svoje aktivnosti vodi v okviru posebnega projekta. V okviru projekta se izvajajo tudi usklajevanja s predstavniki uporabnikov sistema iz razvojnih področij, servisnega, prodajnega in drugih področij. Odločitev ni le tehnološke narave in v poslovnem smislu ne bo lahka, saj bo predvidoma zadevala naložbo, ki se meri v milijonih ameriških dolarjev. Neposredno bo zadevala okrog 250 sodelavcev v področjih za razvoj, uvajanje in vzdrževanje proizvodov SI2000. Glede na izkušnje lahko govorimo o več nivojih sprejemanja odločitve:

- izvedba tehničnih in poslovnih analiz, ovrednotenje in primerjave sistemov ter izdelava predloga glede izbire najprimernejšega sistema;
- iskanje konsenza z vodstvom poslovne enote, srednjim managementom, ključnimi uporabniki in internimi strokovnjaki, ki imajo na tem področju ustrezno znanje in izkušnje;
- priprava, izvedba in preučitev rezultatov pilotskega projekta, z namenom praktičnega preizkusa sistema v realnem okolju;
- sprejemanje končne odločitve na ravni celotne poslovne enote.

Pri končni odločitvi bo imela med drugim pomembno vlogo tudi obstoječa strategija razvoja informacijskega sistema v podjetju in njegovi poslovni enoti ITWE.

1.4.2 Sodelavci v odločitveni skupini

Namen odločitvene skupine je, na podlagi izdelanih teoretičnih ocenjevanj CM sistemov in praktičnih preizkusov sistemov v pilotskih projektih, izdelati predlog rešitve ter sodelovati pri odločitvi, kateri podporni sistem za upravljanje konfiguracij proizvodov uporabljati v naslednjih nekaj letih. Sestavljajo jo vodstvo in sodelavci s področja za tehnično informatiko ter predstavniki ključnih skupin uporabnikov obstoječih podpornih sistemov.
Skupina o tekočih aktivnostih redno obvešča vodstvo oziroma direktorja poslovne enote ITWE. V njej se izvaja velika večina ključnih razvojnih projektov za izdelke SI2000.

1.5 Metoda dela

Pri izbiri sistema za upravljanje konfiguracij izdelkov sem izbral metodo večparametrskega odločanja s pomočjo ekspertnega sistema DEX. Za razvrščanje ovrednotenih variant je bila izbrana metoda portfelja, pri čemer sta bila upoštevana ocena konkurenčne prednosti posameznega podpornega sistema in ocena izpolnjevanja zahtev podjetja. Značilnosti metode in ekspertnega sistema so predstavljene v nadaljevanju.

1.5.1 Elementi uporabljene metodologije

1.5.1.1 Kaj je odločitveni problem

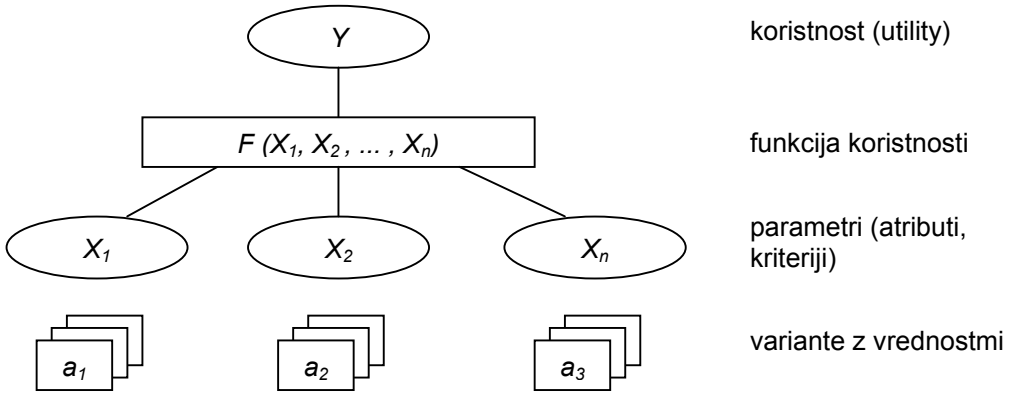
Odločitveni problem lahko v splošnem definiramo takole (Bohanec, 1991, str. 2): dani sta množica ciljev in množica možnosti, določeni s strani izvajalca odločitve. Poiskati je potrebno možnost, ki najbolje ustreza ciljem, ali razvrstiti možnosti od najboljše do najslabše glede na dani cilj. Pri tem naj bodo možnosti (tudi: alternative, variante) objekti ali akcije enakega ali podobnega tipa, npr. različni računalniški sistemi, različne poslovne strategije ipd.
Problemi te vrste se pojavljajo na mnogih področjih človekovega udejstvovanja. Z namenom podpore procesu sprejemanja odločitev je bilo razvitih več metod in računalniških programov. En od pristopov, ki se široko uporablja v praksi, je večparametrsko odločanje (angl. multi-attribute decision making) s pomočjo ekspertnega sistema DEX.

1.5.1.2 Večparametrsko odločanje

Glavna ideja *večparametrskega odločanja* je razgradnja odločitvenega problema v manjše in lažje obvladljive podprobleme (Bohanec, 1991, str. 2). Variante razčlenimo na posamezne parametre (kriterije, atribute). Variante ocenjujemo z vrednotenjem njihovih parametrov. Ti se vrednotijo neodvisno. Oceno variante dobimo s postopkom združevanja ocen parametrov - običajno govorimo o t.im. *funkciji koristnosti.* Končna izpeljana vrednost predstavlja temelj za razvrščanje variant (Bohanec, Rajkovič, 1995).
Vrednotenje variant poteka na osnovi *večparametrskega odločitvenega modela*. Ta je v splošnem sestavljen iz treh komponent (Bohanec, Rajkovič, 1995):

- vhod v model predstavljajo kriteriji X_i. To so spremenljivke, ki ponazarjajo podprobleme odločitvenega problema, to je tiste dejavnike, ki opredeljujejo kakovost variant;
- funkcija koristnosti F je predpis, po katerem se vrednosti posameznih parametrov združujejo v spremenljivko Y, ki ponazarja končno oceno ali koristnost variante;
- variante opišemo po osnovnih parametrih z vrednostmi a_i. Na podlagi teh vrednosti funkcija koristnosti določi končno oceno vsake variante.

Slika 1: Večparametrski odločitveni model



Vir: Bohanec, Rajkovič, 1995

1.5.1.3 Pristop k večparametrskemu odločanju s sistemom DEX

DEX je lupina ekspertnega sistema, namenjena podpori modeliranja večparametrskih odločitev. Njegov glavni namen je (Bohanec, 1991, str. 1) podpora osebi, ki opravlja zapletene večparametrske odločitve, kot npr. izbira vrste tehnologije, izbira kandidata za določeno delo,

nakup hiše ipd. Za tovrstne odločitve, ki se pogosto pojavljajo v vsakdanjem življenju, je značilno, da je potrebno upoštevati veliko možnosti, z njihovimi dobrimi in slabimi stranmi. Te je potrebno preučiti, ovrednotiti in primerjati drugo z drugo. DEXi je verzija sistema DEX, ki je nastala pred kratkim in je namenjena predvsem študentom (Bohanec, Rajkovič, 1999, str. 491). Načrtovana je razširitev v funkcionalno popoln profesionalni sistem DEX.
Ob uporabi sistema DEX uporabnik sam razvije "odločitveni prostor". To stori z določitvijo atributov in njihove ravni. Pri tem v sistem sam vgradi svoje znanje v zvezi z odločitvijo. To stori z navedbo preprostih ugotovitev, imenovanimi tudi *elementarna odločitvena pravila*, kot npr.: "Če je cena visoka, kakovost pa nizka, je ta možnost nesprejemljiva."
V DEX-u se večparametrski pristop k odločanju povezuje z nekaterimi elementi ekspertnih sistemov in *strojnega učenja* (angl. machine learning) (Bohanec, 1991, str. 3). Struktura atributov in funkcije koristnosti se obravnavajo kot eksplicitna baza znanja, sestavljena iz enega ali večih dreves atributov, funkcij koristnosti in opisa variant. Vendar pa obstaja nekaj razlik s konvencionalnimi ekspertnimi sistemi, npr.: v običajnih ekspertnih sistemih so funkcije koristnosti opisane z neko funkcijo, npr. z uteženo vsoto parametrov. V DEX-u so te funkcije določene s preprostimi pravili (odločitvena pravila); za sistem DEX so značilni transparentnost, razumljivost in razložljivost baze znanja in dobljenih rezultatov vrednotenja, medtem ko konvencionalni ekspertni sistemi običajno delujejo kot "črne škatle" (angl. black box); itd. Poleg tega je DEX lupina ekspertnega sistema, kar pomeni, da ne vsebuje vnaprej določene baze znanja. Uporabnik uporablja lastno bazo znanja, povezano z obravnavanim problemom.
Izkušnje kažejo, da se metodologija DEX dobro obnese predvsem v zvezi z "mehkimi", t.j. manj strukturiranimi in manj formaliziranimi odločitvenimi problemi (Bohanec, Rajkovič, 1999, str. 487). Pokazalo se je, da je uporabnost sistema DEX večja, bolj kot je odločitveni problem kompleksen, težaven (Bohanec, Rajkovič, 1999, str. 490). Manj pa je uporaben v primerih, ki zahtevajo natančne formalne modele ali numerične simulacije in optimizacije.

1.5.2 Ključni koraki pri izvajanju odločitev s pomočjo sistema DEX

Odločitveni proces je proces sistematičnega zbiranja in urejanja znanja. Zagotovil naj bi dovolj informacij za primerno odločitev, zmanjšal možnost, da bi kaj spregledali, pohitril in pocenil proces odločanja ter dvignil kakovost odločitve (Bohanec, Rajkovič, 1995). Z DEX-om gre uporabnik skozi različne faze tega procesa. Pri tem so dovoljena iterativna vračanja na prejšnje stopnje procesa, v skladu z naraščanjem uporabnikovega znanja v zvezi z odločitvenim problemom (Bohanec, 1991, str. 3).

1.5.2.1 Identificiranje problema

Ta faza je rezultat spoznanja, da je nastopil odločitveni problem, ki je dovolj težak, da ga je smiselno reševati na sistematičen in organiziran način. V tej fazi poskušamo določiti problem (npr.: kateri avto kupiti) ter opredeliti cilje in zahteve (Bohanec, Rajkovič, 1995). Pri tem oblikujemo odločitveno skupino, katere jedro sestavljajo odločevalci (t.i. "lastniki problema"): to so tisti, ki se morajo v končni fazi odločiti in so odgovorni za odločitev. Poleg tega naj bi hkrati s tem določili vsaj nekaj ključnih lastnosti oziroma atributov, ustrezajočih danemu problemu. Ta korak se običajno izvede brez sistema DEX (Bohanec, 1991, str. 4).

1.5.2.2 Identificiranje in strukturiranje kriterijev

V tej fazi določimo kriterije, na podlagi katerih bomo ocenjevali različice. Zasnujemo strukturo odločitvenega modela (Bohanec, Rajkovič, 1995). Pomembno je, da pri tem ne spregledamo kriterijev, ki bistveno vplivajo na odločitev (načelo polnosti). Pri oblikovanju modela poskušamo

izpolniti tudi nekatere druge zahteve, kot so strukturiranost, neredundantnost, ortogonalnost in merljivost kriterijev. Postopek identificiranja kriterijev je do neke mere odvisen od uporabljene metodologije. Pogosto poteka po naslednjih korakih:

- *seznam kriterijev* - sami ali med pogovorom v skupini oblikujemo nestrukturiran seznam kriterijev, ki jih bomo upoštevali pri odločanju;
- *hierarhično strukturiranje kriterijev* - kriterije hierarhično uredimo, upoštevajoč medsebojne odvisnosti. Nepomembne kriterije in tiste, ki so izraženi z ostalimi kriteriji, zavržemo in po potrebi oblikujemo nove. Rezultat je drevo kriterijev. Na podlagi izkušenj se je izkazalo, da je to najtežji del procesa uporabe metodologije DEX. Ta stopnja je odvisna od znanja in izkušenj izvajalcev odločitev in zahteva dobro razumevanje odločitvenega problema (Bohanec, Rajkovič, 1999, str. 490);
- *zaloge vrednosti* - vsem kriterijem v drevesu določimo zaloge vrednosti, ki jih lahko zavzamejo pri vrednotenju, ter morebitne druge lastnosti (npr. urejenost). Tipični primer za vrednosti iz domene atributa so npr. "majhen", "srednji", "velik". Razen besednih opisov vrednosti iz domene se lahko uporabijo tudi diskretni številski opisi (npr. za število vrat pri avtomobilu) ali številski intervali (npr. za višino naložbe).

1.5.2.3 Določitev funkcij koristnosti

V tej fazi definiramo funkcije, ki opredeljujejo vpliv nižjenivojskih kriterijev na tiste, ki ležijo višje v drevesu, vse do korena drevesa, ki predstavlja končno oceno variant (Bohanec, Rajkovič, 1995). Od izvajalca odločitve se zahteva vnos elementarnih odločitvenih pravil (Bohanec, 1991, str. 4) tipa "Če ... potem ...". Oblika funkcij in način njihovega zajemanja je odvisna od uporabljene metode. Najpogosteje se uporabljajo preproste funkcije, kot so utežena vsota in razna povprečja, srečamo pa tudi zahtevnejše funkcije, ki imajo večjo izrazno moč, vendar so nekoliko zahtevnejše za praktično uporabo: funkcije zvezne logike, funkcije na osnovi Bayesovega pravila ali mehkih množic itd. Orodje pri tem uporabniku učinkovito pomaga in ga vodi pri delu, npr. s spremljanjem konsistentnosti pri določanju funkcije koristnosti. Ta in naslednje faze se običajno izmenično ponavljajo in omogočajo učinkovit proces izvajanja analize in učenja v zvezi z danim odločitvenim problemom (Bohanec, 1991, str. 4).

1.5.2.4 Opis variant

Vsako varianto opišemo z vrednostmi osnovnih kriterijev. Do tega opisa nas vodi proučevanje variant in zbiranje podatkov o njih (Bohanec, Rajkovič, 1995). Pri tem se pogosto srečamo s pomanjkljivimi ali nezanesljivimi podatki. Nekatere metode v tem primeru odpovedo, druge pa omogočajo, da takšne podatke opišemo v obliki intervalov ali verjetnostnih porazdelitev.

1.5.2.5 Vrednotenje in analiza variant

Vrednotenje variant je postopek določanja končne ocene variant na podlagi njihovega opisa po osnovnih kriterijih. Vrednotenje poteka "od spodaj navzgor", v skladu s strukturo kriterijev in funkcij koristnosti (Bohanec, Rajkovič, 1995). V kolikor je pri tem opažena kakršna koli neskladnost, tako glede načina določanja modela kot glede dobljenih rezultatov, je uporabniku omogočeno enostavno vračanje nazaj v fazi definiranja funkcije koristnosti in opisa variant (Bohanec, 1991, str. 6). Varianta, ki dobi najvišjo oceno, je praviloma najboljša (besedo "praviloma" je potrebno na tem mestu poudariti). Na končno oceno vpliva mnogo dejavnikov in pri vsakem od njih lahko pride do napake. Poleg tega sama končna ocena navadno ne zadostuje za celovito sliko o posamezni varianti. Variante je potrebno analizirati in poskusiti odgovoriti na vrsto vprašanj, kot npr.:

- Ali so vrednosti kriterijev in uporabljene funkcije koristnosti ustrezni? Zakaj je končna ocena takšna, kot je? Je v skladu s pričakovanji ali odstopa in zakaj?
- Kakšna je občutljivost odločitve: kako spremembe vrednosti kriterijev vplivajo na končno oceno? Ali je mogoče in kako variante izboljšati?
- V čem se variante bistveno razlikujejo med seboj?

Šele tako pridemo do celovite slike o variantah in s tem do kvalitetnejše in bolje utemeljene odločitve. Računalniška podporna orodja so pri tem praktično nepogrešljiva, saj tovrstne analize s svojimi pripomočki bistveno olajšajo.

2. Področje upravljanja s konfiguracijo izdelkov

2.1 Pomen upravljanja s konfiguracijo izdelkov

2.1.1 Kaj je upravljanje s konfiguracijami in čemu je namenjeno

Konfiguracijo izdelka v splošnem sestavljajo gradniki oziroma sestavni deli izdelka, najsi bodo to funkcije proizvoda ali običajne datoteke. Glede na računalniško obliko, vsebovano informacijo, način njihove obdelave v procesu itd., gradnike lahko razvrstimo v različne tipe ali skupine. *Upravljanje konfiguracij izdelkov* je disciplina za organiziranje in nadzorovanje nastajajočih proizvodov. Izvor ima v letalski industriji v 50. letih 20. stoletja.

Upravljanje konfiguracij (angl. configuration management, krat. CM) se v praksi v splošnem ne izvaja le za končne proizvode in njihove komponente, običajno se obravnava kot splošnejša disciplina. Vključuje lahko kakršne koli elemente v zvezi z izdelki in procesi (Frey-Pučko, 2002, str. 3).

Definicij in razumevanj upravljanja konfiguracij je tako v praksi kot v literaturi zelo veliko. Obstaja cela vrsta razmeroma splošnih in intuitivnih definicij, npr. (Lyon, 2001, str. 18): "upravljanje konfiguracij je proces upravljanja sprememb" ali "upravljanje konfiguracij je disciplina upravljanja, ki se uporablja za zajemanje in nadzor podatkov o izdelkih". Prva od pravkar navedenih definicij temelji na dejstvu, da je proces upravljanja konfiguracij običajno neločljivo povezan s procesom upravljanja sprememb.

Celovitejše definicije upravljanja konfiguracij lahko najdemo v industrijskih, vojaških in drugih standardih v zvezi z upravljanjem izdelkov in njihovih komponent, npr. v MIL-STD-973 (Lyon, 2001, str. 18)*:* "Z ozirom na gradnike konfiguracije je to disciplina, ki uporablja tehnične in administrativne postopke v toku življenjskega cikla tega gradnika, z namenom:

- poistovetiti in dokumentirati funkcionalne in fizične lastnosti gradnikov konfiguracije;
- nadzorovati spreminjanje gradnikov konfiguracije in z njimi povezane dokumentacije;
- obdelati informacije, potrebne za učinkovito upravljanje gradnikov konfiguracije, vključujoč stanje izvajajočih se sprememb;
- presoditi gradnike konfiguracije, z namenom preveriti skladnosti s specifikacijami, tehničnimi risbami, dokumenti za nadzor vmesnikov in drugimi zahtevami iz pogodb.

Z ozirom na podatkovne datoteke v digitalni obliki je to disciplina, ki uporablja pravila poistovetenja in vodenja stanj izbrane konfiguracije, z namenom:

- enolično prepoznati podatkovne datoteke, vključno z verzijami in stanji teh datotek;

- zajemati in poročati o informacijah, potrebnih za učinkovito upravljanje podatkovnih datotek, vključno s stanjem ažuriranih verzij datotek."

Večina definicij v strokovni literaturi in različnih standardih temelji na splošno sprejeti ugotovitvi, da je upravljanje konfiguracij po svojem poslanstvu *proces*. Pri tem se največkrat navajajo naslednje ključne poddiscipline oziroma podprocesi upravljanja konfiguracij, vključeni tudi v zgoraj omenjeno definicijo: poistovetenje (identificiranje) konfiguracij; nadzor konfiguracij; vodenje stanj konfiguracij; presojanje konfiguracij. Z izvajanjem navedenih poddisciplin oziroma podprocesov naj bi zagotovili (Kelly, 1996, str. 13), da:

- vemo, kaj bomo proizvajali, iz katerih sestavnih delov oziroma gradnikov bomo to proizvedli in v kakšnem stanju so le-ti;
- le pravi ljudje lahko uporabljajo in spreminjajo te gradnike in da razumejo vpliv načrtovanih in izvedenih sprememb;
- so dostopna uporabna poročila;
- se izvajajo sporazumno sprejeti postopki.

Posamezne poddiscipline upravljanja konfiguracij lahko razdelimo naprej na posamezne funkcije in sklope funkcij - glej tabelo 1.

Tabela 1: Poddiscipline upravljanja konfiguracij in njihove funkcije

Poddisciplina	Funkcije in sklopi funkcij
Poistovetenje (identificiranje) konfiguracij	• Planiranje posameznih gradnikov in struktur v konfiguraciji • Dogovori v zvezi s poimenovanjem gradnikov • Zagotovitev podatkovnih struktur in spominskega prostora za shranjevanje podatkov in metapodatkov o gradnikih konfiguracije • Načini oštevilčenja verzij in nalog • Planiranje osnovnic in izdaj • Določitev oštevilčenja obrazcev, namenjenih nadzoru konfiguracij
Nadzor konfiguracij	**Nadzorovano področje ali odlagališče podatkov (knjižnica):** • Rezervacija in izdajanje gradnikov v knjižnico • Nadzor stanj in verzij gradnikov • Nadzor porazdelitve gradnikov **Poročanje o problemih/napakah:** • Preiskave v zvezi z morebitnimi problemi na gradnikih • Razčiščevanje problemov in vzpostavljanje ustrezne strukture **Nadzor sprememb:** • Analiza vplivov sprememb gradnikov na strukturo in na druge gradnike • Pooblaščena popravila gradnikov • Pregledi in testiranja
Vodenje stanj	• Zbiranje podatkov (metrike) • Izdelava poročil • Analize podatkov
Presoje konfiguracij	• Zagotovitev izvajanja postopkov upravljanja konfiguracij v skladu s planom • Preverjanje metapodatkov o gradnikih in strukturi

Vir: Kelly, 1996, str. 15

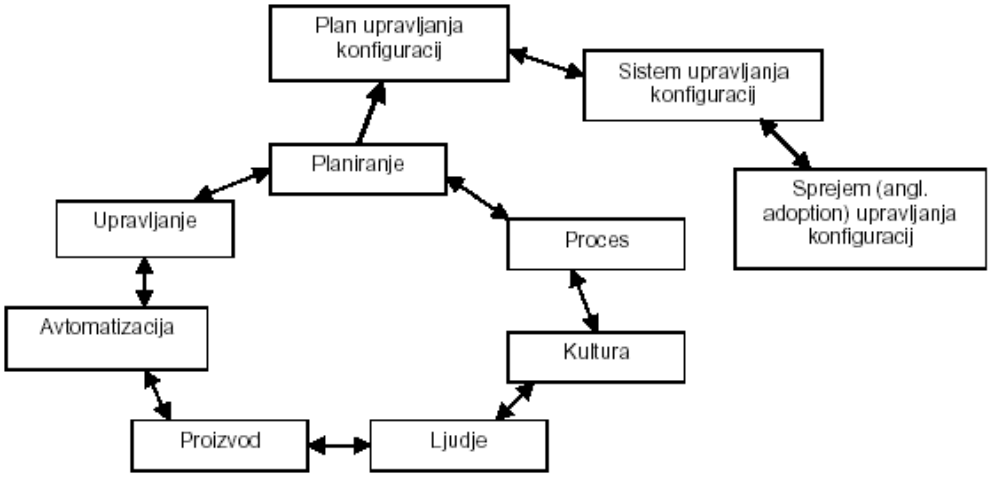
Eno bistvenih poslanstev procesa upravljanja konfiguracij je v zagotovitvi sledljivosti in ponovljivosti procesov v življenjskem ciklu izdelkov. Upravljanje konfiguracij, z zgoraj navedenimi poddisciplinami, je povezovalno sredstvo, ki *zagotavlja strukturo in s tem stabilen temelj za upravljanje podatkov o izdelkih* (CMstat, 2002a) skozi nenehne in sočasno se izvajajoče spremembe. Upravljanje konfiguracij proizvodov ima svojo vlogo tudi pri vzdrževanju konsistentnosti med komponentami izdelka (PO, SO...) in podatki o spremembah skozi ves življenjski cikel proizvoda (Taramaa, 1998, str. 47).
Upravljanje konfiguracij izdelkov lahko z ozirom na naravo obravnavanih gradnikov projiciramo na več posebnih disciplin, npr. na *upravljanje konfiguracij PO* (angl. software configuration management, krat. SCM), ki obravnava proces upravljanja konfiguracij za PO, in *upravljanje konfiguracij SO* (angl. hardware configuration management, krat. HCM), kjer se poleg procesa upravljanja konfiguracij strojne opreme (npr. mehanske komponente) večkrat prišteva tudi upravljanje konfiguracij strojno-programske opreme (npr. tiskana vezja).

2.1.2 Elementi upravljanja konfiguracij in pomen planiranja

Različni analitiki nadalje razčlenjujejo upravljanje konfiguracij izdelkov na osnovne elemente, ki naj bi reševali ključne zahteve za to področje v organizacijah, npr. (Bounds, Dart, 1993, str. 6):

1. določanje ključnih nalog, ki morajo biti dokumentirane v planu upravljanja konfiguracij;
2. opis procesa upravljanja konfiguracij in raven njegovega nadzora;
3. vloge in odgovornosti za naloge, ki se izvajajo v procesu upravljanja konfiguracij;
4. razumevanje in upoštevanje obstoječe kulture podjetja;
5. določanje, kateri izdelki in njihove komponente bodo nadzorovani z metodami upravljanja konfiguracij in kateri deli dejansko sestavljajo izdelek;
6. določanje funkcionalnih zahtev za avtomatizirani sistem za podporo upravljanju konfiguracij;
7. odločitve vodstva o tem, ali kupiti ali sami razviti sistem za podporo upravljanju konfiguracij ter odločitve o vpeljavi in začetku uporabe sistema.

Slika 2: Elementi upravljanja konfiguracij



Vir: Bounds, Dart, 1993, str. 6

Naslednji trije elementi naj bi predstavljali ključ do uspešne rešitve na področju upravljanja konfiguracij in so izpeljani iz zgoraj navedenih osnovnih elementov (Bounds, Dart, 1993, str. 6):

1. plan za upravljanje konfiguracij - namenjen izpolnitvi zahtev v zvezi z upravljanjem konfiguracij;

2. sistem za upravljanje konfiguracij - orodja, izbrana za podporo avtomatiziranih aktivnosti procesa upravljanja konfiguracij;
3. strategija uveljavljanja upravljanja konfiguracij - strategija, ki organizaciji omogoča, da proces in sistem upravljanja konfiguracij dobita ustrezen pomen v poslovnem okolju.

Planiranje izvajanja upravljanja konfiguracij ni del definicije le-tega. Je pa en njegovih najpomembnejših elementov, ki zaradi zapletenosti izdelkov in procesa postaja vse pomembnejši. Plan naj bi vseboval ne le opis odgovornosti in vlog v procesu, temveč tudi (Angstadt, 2000, str. 27): uporabljene metrike; razpoložljiva in potrebna znanja; infrastrukturo; analizo možnih dogodkov in situacij ter predvideni ukrepi ob njihovem pojavu; spremljanje vložkov in poteka dela; pogodbe in odgovornosti v zvezi z njimi; nadzor in načine nadziranja; spremljanje porabljenih virov (npr. za šolanja); definicijo procesa, itd.

2.1.3 Proces upravljanja konfiguracij in poslovno okolje

2.1.3.1 Vloga procesa upravljanja konfiguracij v poslovnem okolju

Upravljanje konfiguracij se uporablja kot poslovni proces v mnogih organizacijah, med drugim v domala vseh podjetjih, ki se ukvarjajo z razvojem in vzdrževanjem izdelkov, ki vsebujejo kompleksnejšo PO (CMstat, 2002). Omenjeni proces je en ključnih dejavnikov uspešnega upravljanja in nadziranja zelo kompleksnih projektov, v katerih se razvija in/ali vzdržuje PO (Burrows, 1999, str. 12). Dosežena raven upravljanja konfiguracij predstavlja eno od temeljnih zagotovil za resnost pristopa k izpeljavi poslov, ki se izvajajo z razvojnimi projekti.

Ena ključnih prednosti uporabe tega procesa je odkrivanje in odpravljanje napak proizvoda v zgodnjih fazah življenjskega cikla. Rezultati različnih raziskav kažejo, da je popravljanje komponente izdelka po izročitvi naročniku deset do dvajsetkrat dražje kot popravljanje v fazi kodiranja oziroma realizacije (Leffingwell, Widrig, 2000, str. 10). Strategija zmanjševanja stroškov za proizvode naj bi bila sploh ena najučinkovitejših strategij podjetij za doseganje njihove poslovne uspešnosti (Omlie, 2000, str. 14). Pri tem je pomembno, da proizvod in denarne tokove, povezane z njim, obravnavamo skozi celotni življenjski cikel izdelka. Ne smemo se omejiti le na obravnavanje razvojnih projektov. Šele s celovitim upravljanjem proizvodov skozi ves njihov življenjski cikel dobi upravljanje konfiguracij svoj pravi pomen.

Posebej aktualna je podpora upravljanju konfiguracij za geografsko porazdeljene poslovne sisteme. V preteklih letih je bilo na tem področju izvedenih precej raziskav, ki dokazujejo, da je izvajanje upravljanja konfiguracij v takšnih sistemih izvedljivo (Milewski, 1997, str. 105).

Pri mnogih najuspešnejših podjetjih se opaža težnja k vse bolj celoviti obravnavi upravljanja konfiguracij. To med drugim pomeni, da ustrezne pristope in sisteme uporabljajo v celotnem poslovnem sistemu in ne le v omejenem obsegu, na nivoju določenega oddelka ali področja. Pri tem upravljanja konfiguracij ne uporabljajo le za sledenje in poročanje o poteku procesov načrtovanja, proizvodnje in podpore izdelkov, temveč kot enega od strateških pristopov k upravljanju (CMstat, 2002a). Na redni letni konferenci združenja CMII junija 2000 v Minneapolisu je bilo npr. na neformalni ravni opozorjeno na občutljivo povezavo med upravljanjem konfiguracij izdelkov in komunikacijo s kupcem, kar ima lahko posledice na pravnem področju. Proizvajalec naj bi vnaprej jasno sporočil kupcu izdelka tako sestavo kot namen in uporabnost izdelka. V kolikor nato uporabnik tega ne upošteva in pri uporabi pride do nepričakovanih posledic uporabe, praviloma sam nosi posledice. Le v primeru, ko ta dosledno upošteva vse informacije proizvajalca, pa je kljub temu prišlo do nepričakovanega pripetljaja, lahko pravne posledice zadevajo tudi proizvajalca danega izdelka. Omenjen primer kaže, da upravljanje konfiguracij izdelkov lahko vpliva na zelo širok spekter ravni poslovanja podjetij. V

nadaljevanju so opisane nekatere najbolj značilne povezave procesnega področja upravljanja konfiguracij z nekaterimi drugimi procesnimi področji v industrijskih podjetjih.

2.1.3.2 Povezanost upravljanja konfiguracij z drugimi procesnimi področji

Proces upravljanja konfiguracij izdelkov je tesno povezan ne le z inženirskim procesom načrtovanja, realizacije in podpore izdelkov, temveč tudi z naslednjimi procesi: upravljanje sprememb proizvodov; proces planiranja in vodenja projektov; proces zagotavljanja kakovosti; upravljanje zahtev za izdelke.

Na področju programske opreme je uspešno izvajanje pravkar navedenih procesnih področij hkrati z upravljanjem konfiguracij pogoj za doseganje drugega nivoja zrelosti razvojnega procesa po CMM modelu. Skupni imenovalec za drugi nivo CMM modela je zagotavljanje sledljivosti v razvojnem procesu. Brez ustrezne ravni upravljanja konfiguracij ne bi bilo mogoče zagotoviti uspešnega izvajanja navedenih procesov.

2.1.3.2.1 Povezanost z upravljanjem sprememb proizvodov

Spremembe se dogajajo neprenehoma in vključujejo tako spremembe v načinu poslovanja kot spremembe v tehnologiji razvojnih in podpornih okolij ter nenazadnje na končnih izdelkih (Stevens, 2000, str. 26). Procesa upravljanja konfiguracij in upravljanja sprememb nad izdelki sta nerazdružljivo povezana (Taramaa, 1998, str. 60). Medtem ko upravljanje konfiguracij nadzira statični vidik strukture izdelkov, vključno z njihovimi verzijami in izhodiščnimi konfiguracijami, upravljanje sprememb izdelkov nadzira dinamični vidik, t.j. izvedbo sprememb nad temi strukturami.

Kot en od pogojev za učinkovito upravljanje konfiguracij proizvodov mora biti zagotovljen nadzor nad spremembami vseh identificiranih elementov, ki jih bomo nadzorovali v procesu upravljanja konfiguracij (Frey-Pučko, 2002, str. 3). Nadzor mora zagotavljati sledljivost nad vsemi tipi gradnikov v različnih konfiguracijah, odobritev novih konfiguracij in sprotno ažuriranje. Podobno, kar velja za celotne proizvode, velja tudi za proces upravljanja konfiguracij PO.

Problemi s spreminjanjem PO, ki povzročajo največ preglavic, so običajno povezani s slabim upravljanjem konfiguracij (Humphrey, 1989, str. 113). Primeri tovrstnih problemov: težka napaka, odpravljena z velikimi težavami, se ponovno pojavi; razvita in stestirana funkcionalnost manjka v končnem izdelku; dobro pretestiran program kar naenkrat več ne dela, ipd.

2.1.3.2.2 Povezanost z vodenjem projektov in zagotavljanjem kakovosti

Upravljanje konfiguracij proizvodov je pomemben element planiranja in vodenja razvojnih projektov. Obe disciplini imata precej skupnih zahtev in metodologij, med katerimi so (Lyon, 2001, str.109): pogajanja in prilagajanje zahtev; zahteva za učinkovito komunikacijo; planiranje projektnih programov; odvisnost od formalnih urnikov; potreba po zgodnji zaznavi morebitnih problemov, vključno z izvedbo korektivnih ukrepov.

Aktivnosti upravljanja konfiguracij zagotavljajo večjo preglednost nad potekom projekta (Lyon, 2001, str. 23), tako za funkcionalni kot za projektni management. Večja preglednost pa je nenazadnje pomembna tudi za kupca. Plan upravljanja konfiguracij, vključujoč scenarije in razporede vhodov in izhodov za aktivnosti v procesu upravljanja konfiguracij, zagotavlja orodje, s katerim lahko sledimo stanju proizvoda skozi njegov celotni življenjski cikel in zagotovimo, da se pogodbene aktivnosti in dogodki v projektih izvajajo v skladu z dogovorom oziroma planom. Pri izvajanju operativnih aktivnostih je pomembno, da oddelki, ki planirajo razvojne in proizvodne aktivnosti v projektu, upravljanje konfiguracij in zagotavljanje kakovosti za dani projekt, sodelujejo in podpirajo drug drugega, v skladu s pogodbo in cilji projekta (Lyon, 2001, str. 24). Delovanje teh skupin je med seboj zelo soodvisno, prepleteno.

2.1.3.2.3 Povezanost z upravljanjem zahtev za proizvode

Upravljanje zahtev za proizvode je en najpomembnejših procesov v zvezi z upravljanjem življenjskega cikla proizvodov. Začne se z določitvijo zahtev in nadaljuje skozi projekt, pri čemer izpolni svoje poslanstvo v sprejemljivosti izdelka ali storitve glede na postavljene zahteve (Stevens, 2000, str. 44). Projekt, ki izpolnjuje zahteve, je po definiciji uspešen (Stevens, James, 2002, str. 1).

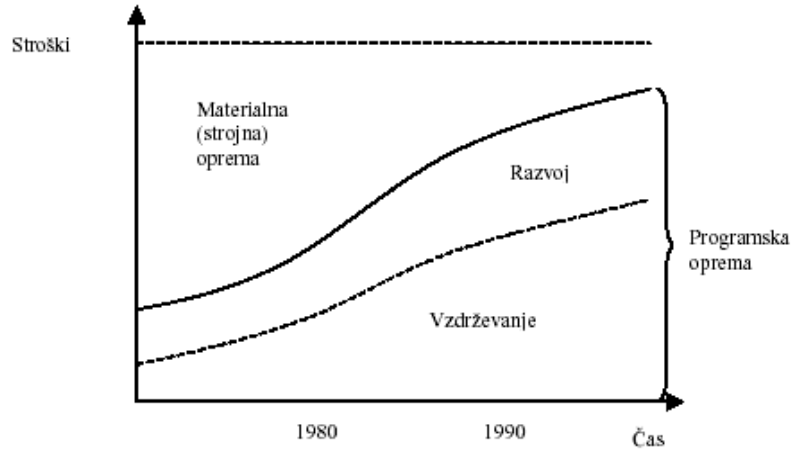
Aktivnosti dobro postavljenega procesa upravljanja konfiguracij izdelkov se v praksi običajno začno že pred začetkom razvojnega projekta, vsaj v fazi identificiranja in strukturiranja zahtev za proizvode. Orodja za upravljanje konfiguracij podpirajo proces upravljanja z zahtevami za proizvode (Stevens, 2000, str. 36) z naslednjimi funkcijami: omogočajo pregledno, strukturirano vodenje zahtev; nadzirajo informacije, povezane z izdajami proizvodov; zagotavljajo sledljivost spreminjanja gradnikov in zahtev; zagotavljajo ponovljivost starih izdaj proizvoda in omogočajo prepoznati, katere zahteve je potrebno izpolniti; podpirajo preverjanje skladnosti komponent izdelka z zahtevami zanje; itd.

Običajno v praksi po fazi specificiranja zahtev za izdelke pride do novih zahtev, ki jih je prej ali slej potrebno implementirati v določeni verziji proizvoda. Upravljanje konfiguracij izdelkov z uporabo ustreznih podpornih sistemov povečuje učinkovitost upravljanja teh *naknadno podanih zahtev* (angl. late features).

2.1.4 Upravljanje konfiguracij za programsko opremo

Ker je PO postala najpomembnejša komponenta telekomunikacijskih izdelkov pa tudi npr. izdelkov s področja letalske, avtomobilske in vojaške industrije (glej sliko 3), kjer tudi vse bolj prevladujejo izdelki z vloženimi računalniškimi sistemi, v nadaljevanju v kratkem ponazarjamo posebnosti upravljanja s konfiguracijami PO (angl. Software Configuration Management, krat. SCM).

Slika 3: Razmerje med ceno programske in strojne opreme v proizvodih



Vir: Pivka, 1996, str. 104

Za proces razvoja PO je značilno, da je "neotipljiv" (Pivka, 1996, str. 18) - večina uporabnikov dobi bolj "otipljive" rezultate šele, ko je proces končan. Tudi sama PO ima nekaj lastnosti, ki jo ločujejo od klasičnih industrijskih izdelkov, sestavljenih predvsem ali izključno iz strojne opreme. PO in SO sta si različni v naslednjih elementih (Berlack, 1992, str. 41): PO je za razliko od SO nevidna; PO ima večjo logično kompleksnost in poleg logike običajno vsebuje tudi podatke; obseg PO lahko hitro narašča, lahko se jo tudi hitro spremeni. PO z navedenimi lastnostmi (neoprijemljivost, prilagodljivost, možnost hitrega spreminjanja...) v upravljanje konfiguracij

prinaša dodatno kompleksnost. Vendar z metodami upravljanja konfiguracij lahko učinkovito obvladujemo tudi PO, saj se gradniki vodijo na računalniku dostopnih medijih.

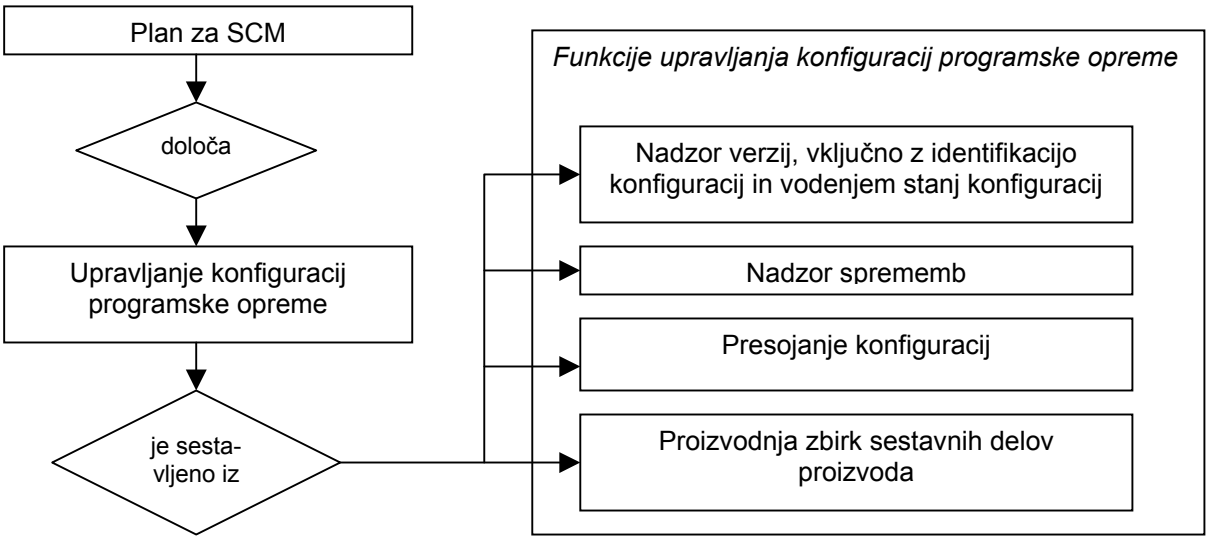
2.1.4.1 Posebnosti upravljanja konfiguracij programske opreme

Upravljanje konfiguracij je ključni element procesa izdelave in vzdrževanja PO (Feiler, 1991, str. 1) in se lahko obravnava v sklopu upravljanja konfiguracij industrijskih proizvodov, sestavljenih iz različnih komponent (PO, SO...).

V splošnem so načela upravljanja konfiguracij tako za programsko kot za strojno opremo enaki (Berlack, 1992, str. 4). Bistvena razlika med upravljanjem konfiguracij za različne tipe gradnikov se kaže le v načinu izvajanja posameznih postopkov (Bounds, Dart, 1993, str. 26) - postopke je namreč potrebno prilagoditi naravi gradnikov. Razlika se npr. pokaže v ravni avtomatizacije postopkov. V primerjavi z upravljanjem konfiguracij klasičnih industrijskih izdelkov je proces upravljanja konfiguracij PO običajno bolje avtomatiziran, saj se razvoj PO izvaja predvsem z računalniško podprtimi okolji (Taramaa, 1998, str. 37).

Definicija za upravljanje konfiguracij PO se v osnovi, v skladu z zgoraj navedenimi ugotovitvami, ujema s splošno definicijo upravljanja konfiguracij. Definicija za SCM je bila pozneje razširjena (Taramaa, 1998, str. 38) na naslednja področja: (a) naloge v zvezi s podporo proizvodnje PO - vključuje ustvarjanje izpeljanih konfiguracij s pomočjo orodij in mehanizmov za gradnjo PO; (b) upravljanje procesa in (c) podporo skupinskega dela. Tako razširjeno upravljanje konfiguracij PO naj bi postalo disciplina, s katero je možno nadzirati tudi spremembe sistemov. Pri tem se uporabljajo naslednje funkcije: poistovetenje komponent konfiguracije; sledenje sprememb; izbiranje verzij gradnikov, komponent, konfiguracij...; izdelava in upravljanje izhodiščnih konfiguracij; proizvodnja PO; upravljanje sočasnega načina dela; izdelava in upravljanje izdaj PO; podpora dostavljanju in namestitvi PO. Klasična, ožja definicija upravljanja konfiguracij PO je naravnana bolj na upravljanje, razširjena (Feilerjeva) definicija pa bolj na podporni, razvojni vidik (Taramaa, 1998, str. 38).

Slika 4: Osnovni elementi upravljanja konfiguracij PO



Vir: Taramaa, 1998, str. 39

Narava procesa upravljanja konfiguracij izdelkov in njihove PO se skozi svoj razvoj ni veliko spremenila, spremenila pa so se razvojna oziroma podporna in delovna okolja, v katerih delajo razvijalci in drugi udeleženci projektov (Angstadt, 2000, str. 26). Čeprav obstajajo komercialne rešitve oziroma podporni sistemi, pa v konkretnih razvojnih okolji obstaja še vrsta področij

oziroma dodatnih dimenzij (Taramaa, 1998, str. 19), za katere je potrebno poiskati oziroma razviti ustrezne rešitve, kot npr. modeli novih pojavnih oblik življenjskih ciklov, vprašanja, povezana z zgradbo sistemov ipd.

2.1.4.2 Dvojna vloga upravljanja konfiguracij programske opreme

Upravljanje konfiguracij PO ima v praksi več ravni (Feiler, Downey, 1990, str. 3). Nekateri poudarjajo vidik upravljanja razvoja in vzdrževanja PO, medtem ko drugi opozarjajo na tehnični oziroma podporni vidik (Feiler, 1991, str. 1):

- kot disciplina nadzora in upravljanja, upravljanje konfiguracij omogoča nadzor nad nastajanjem izdelka. To omogoča s poistovetenjem komponent in sprememb izdelka, z evidentiranjem, ovrednotenjem in nadzorom sprememb ter s pomočjo vodenja zapisov in poročil o zgodovini in stanjih izdelka ter z njim povezanih procesov;
- kot podporna funkcija upravljanje konfiguracij podpira vzdrževanje komponent izdelka, beleži zgodovino spreminjanja izdelka in njegovih komponent, zagotavlja stabilno okolje za izvajanje sprememb na izdelku, podpira proizvodnjo in sestavljanje izdelka iz njegovih komponent in omogoča usklajevanje sočasnega izvajanja sprememb.

Osebje, ki izvaja naloge v zvezi z aktivnostmi procesa upravljanja konfiguracij PO, mora tako po eni strani nadzirati podatke, po drugi pa podpirati delo udeležencev projekta (Angstadt, 2000, str. 26).

V praksi je pogost primer, da razvojni inženirji, ki so prepuščeni sami sebi, delajo mimo sistema za podporo upravljanju konfiguracij. Svoje načrte in druge delovne proizvode vključijo v nadzorovan sistem šele v poznih fazah razvojnega cikla. Tedaj se največkrat pokažejo številni problemi glede njihove ustreznosti. Po drugi strani osebje, zadolženo za upravljanje konfiguracij izdelkov, želi imeti sistemski nadzor nad razvojno dokumentacijo in drugimi delovnimi proizvodi od samega začetka snovanja proizvoda. Tu obstaja možnost za nesporazume. Vendar na področju upravljanja konfiguracij ni splošnih receptov glede ravni strogosti upoštevanja pravil (Lyon, 2001, str. 24). V danem poslovnem okolju je potrebno zagotoviti prilagodljivo, a vendar dovolj strogo in učinkovito izvajanje procesa.

2.1.5 Upravljanje družin proizvodov, ki vsebujejo vložene računalniške sisteme, in njihovih konfiguracij

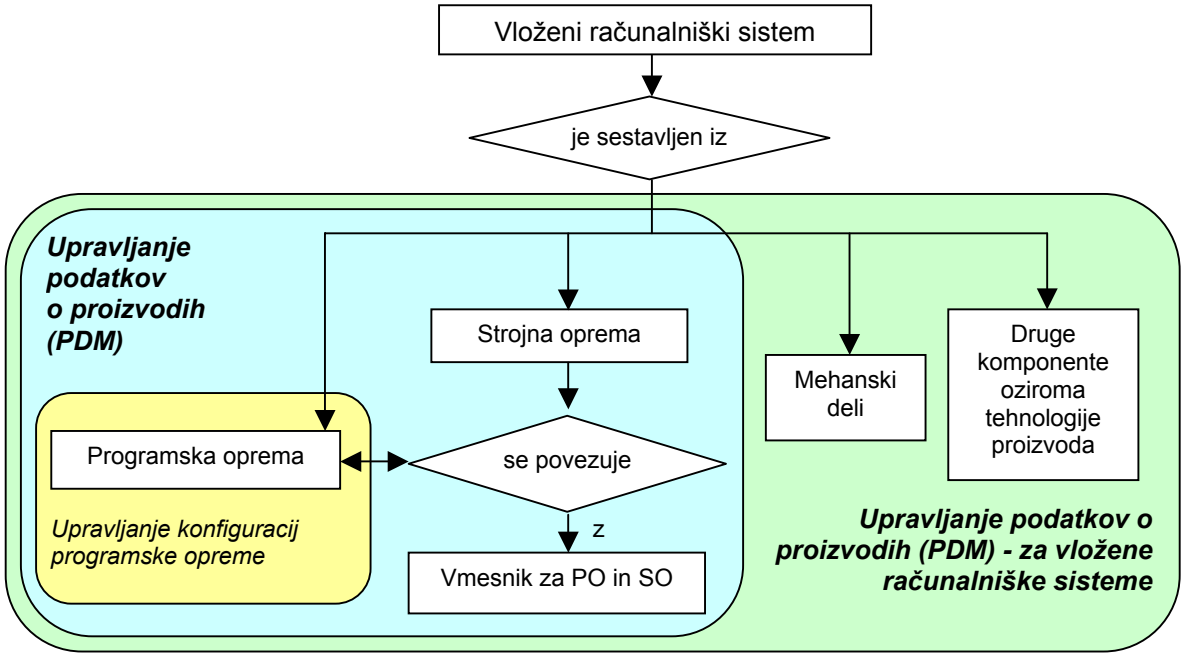
2.1.5.1 Splošne značilnosti

Podjetja s področja telekomunikacijske, vojaške, letalske in drugih vrst industrij se srečujejo s problematiko obvladovanja družin zapletenih izdelkov, katerih ključni sestavni del so vloženi računalniški sistemi. Problem upravljanja konfiguracij družine izdelkov nastane zaradi naslednjega razloga (Ghezzi, Jazayeri, Mandrioli, 1991, str. 406): kot rezultat sprememb ima lahko ena komponenta izdelka različne verzije, izdelek pa lahko doživi več izdaj ali pa se razvije v celo družino podobnih izdelkov. Vsak član družine izdelkov je lahko sestavljen iz različnih verzij komponent. V tovrstnih sistemih se povečuje tako število komponent sistema kot število različnih povezav med njimi - prihaja do t.im. *problema kombinatorične eksplozije* (Asklund et al., 1999, str. 102). Običajna posledica je povečanje števila napak v delovnih in končnih proizvodih, zmanjšanje učinkovitosti komunikacije med člani razvojne skupine ipd. Podjetja so tako prisiljena razviti dosledno metodologijo načrtovanja in razvoja izdelkov, ki mora biti usklajena z obstoječim načinom upravljanja konfiguracij (Lyon, 2001, str. 24). Metodologija običajno temelji na dogodkih, ki omogočajo prehod nadzora z enega nivoja na drugega. Vsi sodelavci v sistemu morajo poznati svoje vloge in odgovornosti ter razumeti, kaj se od njih

pričakuje v posamezni fazi življenjskega cikla izdelka. Zaradi lažjega nadzora večina podjetij uporablja tak ali drugačen sistem mejnikov za izdelke in projekte, ki se vodijo v zvezi s temi izdelki. V praksi so se najbolj uveljavili modeli mejnikov, ki temelje na dogodkih v procesu. Tovrstne modele običajno podjetja prilagodijo zahtevam lastnih projektnih programov. Pri upravljanju izdelkov z vloženimi računalniškimi sistemi je potrebno posebej upoštevati naslednje posebnosti:

- *zmogljivost vloženih računalniških sistemov je vse bolj določena z zmogljivostjo PO.* Medtem ko se stroški in čas za snovanje, razvoj in vzdrževanje PO neprenehoma povečujejo, se vlaganje virov v SO z njeno poenostavitvijo zmanjšuje. Funkcionalnosti se tako največkrat izvajajo v PO, ne v SO (Mäkäräinen, 2000, str. 45). Uporaba enotne strojne platforme, na kateri se izvedejo prilagoditve s pomočjo PO, je običajno cenejši in učinkovitejši način kot načrtovanje različnih verzij SO. Upravljanje različnih verzij prilagojenih proizvodov se tako večinoma prenese na področje upravljanja konfiguracij in sprememb PO. Novi proizvodi namreč navadno temeljijo na izhodiščnih konfiguracijah sistemov in komponent PO, razvitih v prejšnjih projektih. Naraven, običajni način razvoja PO je: izvesti spremembe nad obstoječo PO, jo prilagoditi novi SO, ji dodati nove funkcionalnosti, ne pa je pisati na novo. Pri razvoju in vzdrževanju te PO v splošnem veljajo enake zakonitosti kot pri drugih procesih, v katerih se uporabljajo koncepti programskega inženirstva (Taramaa, 1998, str. 31);
- *tesna povezava med programsko in strojno opremo* v vloženih računalniških sistemih postavlja dodatne zahteve v zvezi z upravljanjem konfiguracij teh sistemov;
- *pogosto se pojavljajoče nove zahteve naročnikov, množenje in diverzifikacija izdelkov* ter njihovih funkcionalnosti prinašajo v razvojni in vzdrževalni proces dodatno zapletenost, ki se zelo odraža na področju upravljanja konfiguracij in sprememb teh izdelkov.

Slika 5: Vloženi računalniški sistemi in upravljanje njihovih komponent



Vir: Taramaa, 1998, str. 48

V planiranju in izvajanju procesa upravljanju konfiguracij za tovrstne sisteme je potrebno upoštevati navedene posebnosti. To npr. pomeni, da se proces upravljanja konfiguracij izdelkov za vložene računalniške sisteme v nekaterih podrobnostih, predvsem na nivoju posameznih

postopkov, razlikuje od upravljanja konfiguracij klasičnih industrijskih izdelkov. Primer: Ker so v vloženih sistemih komponente SO in PO v tesni medsebojni odvisnosti, je potrebno s pomočjo presoj konfiguracij, pri katerih sodelujejo predstavniki obeh strani, določiti ustrezne vmesnike med temi komponentami (Taramaa, 1998, str. 44).
Poleg ustreznega načina upravljanja konfiguracij proizvodov se v praksi kaže, da je za uspešno obvladovanje družin tovrstnih izdelkov potrebno izvajati tudi naslednja vidika upravljanja, ki neposredno vplivata tako na sam inženirski proces kot na upravljanje konfiguracij in sprememb teh izdelkov: upravljanje zgradbe za družino proizvodov in upravljanje produktne linije.

2.1.5.1.1 Upravljanje zgradbe za družino proizvodov

Kakovost načrtovanja sistemov zelo vpliva na kakovost upravljanja konfiguracij. Pomanjkanje znanja glede načrtovanja PO in način načrtovanja brez ustreznega predvidevanja sprememb sta dva od pomembnih dejavnikov, ki povzročata težave na področju upravljanja sprememb PO (Mäkäräinen, 2000, str. 41). Pri obvladovanju zgradbe (arhitekture) in konfiguracij zapletenih proizvodov z vloženimi računalniškimi sistemi prihajata posebej do izraza principa modularnosti in predvidevanja sprememb.

1. *Modularnost ali uporaba modulov, komponent.* Učinkovitost dela na zapletenem sistemu (izdelku), še posebej pa pri pouporabi posameznih funkcij sistema, je v veliki meri odvisna od tega, kako smo bili uspešni pri načrtovanju zgradbe, t.j. pri opisu sistema s podsistemi, moduli, komponentami (Ghezzi, Jazayeri, Mandrioli, 1991, str. 64). Tako se izboljša obvladljivost odvisnosti med različnimi moduli, komponentami izdelka. Pri tem je pomembno, da je vsak od modulov določen na način, ki po eni strani omogoča, da ga z drugimi moduli učinkovito povežemo v celovit sistem, po drugi strani pa mora biti omogočena učinkovita ločena obdelava (razvoj, testiranje, vzdrževanje...) vsakega posameznega modula. Sistemu, ki ga sestavljajo moduli, pravimo, da je modularen. Modularnosti tako omogoča lažjo usmeritev v dva ločena problema:
    - ukvarjanje s podrobnostmi vsakega ločenega modula posebej, pri čemer ločimo podrobnosti v zvezi z vsemi drugimi moduli;
    - ukvarjanje z lastnostmi različnih modulov in povezav med njimi, z namenom povezati jih v enoten sistem. Pri tem module obravnavamo kot sestavne dele obravnavanega sistema.

    Novejše usmeritve pri razvoju PO gredo vse bolj v smeri gradenj sistemov, ki bodo uporabljali pouporabljive komponente PO (Mäkäräinen, 2000, str. 175). Prednost takšnega pristopa ni le v možnosti ponovne uporabe delov sistema, npr. s pomočjo standardiziranih ogrodij (EJB, CORBA, COM+...) in obstoječih komponent, temveč tudi v prožnejši zgradbi sistema, poleg tega pa komponente zagotavljajo naravno osnovo za upravljanje s konfiguracijami (Roubine, 2001, str. 28). Ustrezno obvladovanje zgradbe sistemov s pomočjo modulov oziroma komponent vpliva tudi na boljše možnosti v zvezi z razdelitvijo dela oziroma delovnih nalog (McMahon, 2001, str. 5).
2. *Predvidevanje (anticipacija) sprememb.* Na proizvodih, posebej pa še na njihovi PO, se spremembe dogajajo neprenehoma in zelo pogosto. Glavna razloga sta dva: popravki napak, ki onemogočajo normalno delovanje PO in niso bili odkrite pred izdajo, ter potreba po nadaljnjem razvoju izdelka, izpolnjevanje novih in spreminjajočih se zahtev. To zahteva posebno pozornost v smeri predvidevanja, kako in kje se bodo spremembe verjetno pojavile. V kolikor so verjetne spremembe identificirane že pred ali med fazo načrtovanja, je potrebno zgradbo izdelka zastaviti tako, da bo te spremembe mogoče pozneje čim lažje vgraditi vanj, sam sistem pa se bo lahko nemoteno razvijal in širil (Stevens, 2000, str. 27). En od pristopov temelji na že omenjeni modularizaciji. S tem se spreminjanje omeji na

manjše, bolj obvladljive dele PO. Predvidevanje sprememb zahteva primerna orodja za nadzorovano upravljanje različnih verzij in revizij PO. Podprto mora biti shranjevanje in doseganje dokumentacije ter modulov PO iz osrednjega odlagališča podatkov, v katerem so shranjene pouporabljive komponente izdelka. Celoten sistem mora ostati usklajen, četudi se spreminja le ena od njegovih komponent. Predvidevanje sprememb pa ne zajema le ravni izdelka, temveč tudi procesni vidik. V fazi načrtovanja je potrebno zagotoviti izdelavo takšne zgradbe izdelka, ki bo zagotavljala kakovostnejšo in cenejšo fazo vzdrževanja, ter glede na predvidene spremembe oceniti predvidene stroške in zagotoviti ustrezno organiziranost za optimalno izvedbo teh sprememb (Ghezzi, Jazayeri, Mandrioli, 1991, str. 54).

2.1.5.1.2 Upravljanje produktne linije

Razumevanje potencialnih potreb uporabnikov, iskanje in uresničevanje rešitev ter upravljanje razvoja proizvodov se ne konča pri upravljanju zgradbe in komponent izdelkov. Zahteva sistematičen in celovit pristop. Z učinkovitim upravljanjem produktne linije oziroma celotnega portfelja proizvodov običajno pridemo do metod, s katerimi lahko bolje izrabimo obstoječe vire in zmanjšamo stroške razvoja posameznih proizvodov (Cohen, 2000, str. 19). Pri tem je pomembno identificirati ustrezno, kompetentno osebo, ki bo nase prevzela odgovornost upravljanja in imela ustrezno podporo vodstva.
V zvezi s tem na upravljanje proizvodov in njihovih konfiguracij vplivajo tudi odločitve v zvezi s pravili, na kakšen način razvijati splošne, generične dele komponent izdelka, in na kakšen način specifične, od tržnih dejavnikov odvisne sestavne dele. To med drugim vpliva na način izdelave izhodiščne konfiguracije novih različic izdelkov, npr. s pouporabo že definirane strukture obstoječih izdelkov.

2.1.5.2 Razvojni proces

Kot že omenjeno, struktura vloženih računalniških sistemov vpliva na to, da se spremembe pri tovrstnih sistemih izvajajo na poseben način. Proces razvoja PO vse bolj dobiva naravo procesa upravljanja sprememb PO (Mäkäräinen, 2000, str. 74). Vzroka za to, da večina razvijalcev običajno ne razvija nove PO, temveč jo le spreminja, je predvsem v tem (Mäkäräinen, 2000, str. 40), da je življenjski cikel sistemov dolg, faza vzdrževanja pa običajno precej daljša od razvojne faze. Proizvodi, ki vsebujejo vloženo PO, običajno zahtevajo mnoge obnovitvene cikle. Statistične študije kažejo (Taramaa, 1998, str. 18), da je povprečna življenjska doba tovrstnih sistemov sedem let, obstajajo pa tudi proizvodi, ki se uporabljajo po 20 let in več. Za obstoječe sisteme je potrebno zagotoviti neprekinjeno podporo. Pri tem se zahteve za sisteme neprenehoma spreminjajo. Potrebno jih je izpolnjevati, sicer le-ti zastarijo. Novi sistemi in njihove konfiguracije tako običajno nastanejo iz obstoječih, z dodajanjem novih zahtev. Spremembe na vloženi PO imajo bolj značilnosti vzdrževanja kot razvoja (Taramaa, 1998, str. 20). Pri tem gre v tej fazi največkrat za prilagoditveno in izboljševalno vzdrževanje.

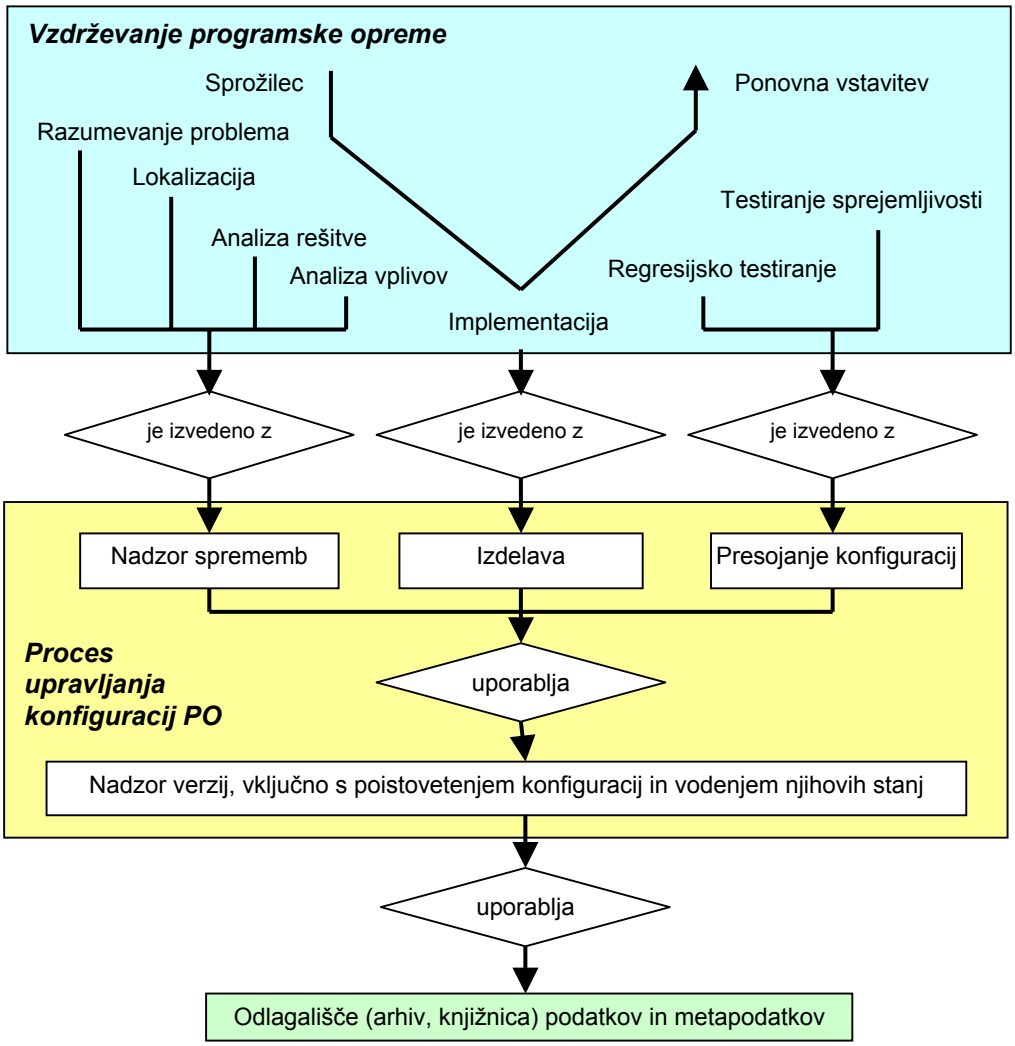
2.1.5.3 Vzdrževanje

Pomen faze vzdrževanja je bil dolgo podcenjen. Splošne kategorije vzdrževanja, t.s.

- prilagoditveno (z namenom prilagajanja spremembam v okolju),
- korektivno (diagnosticiranje in odprava napak na sistemih),
- izboljševalno (spremeniti in izboljšati funkcionalnosti sistema) in
- preventivno vzdrževanje (izboljšati lastnosti PO, tako da jo bo mogoče vzdrževati dalj časa - to je povezano s predvidevanjem sprememb v tehnologiji sistemov),

so bile vpeljane konec sedemdesetih let 20.stoletja. Raziskave iz sredine devetdesetih let so pokazale (Taramaa, 1998, str. 49), da vzdrževanje PO pobere 40 do 70% stroškov za PO. Raziskave iz istega časa kažejo, da je okrog 20% vzdrževanja povezano z odpravljanjem napak, medtem ko je preostali del namenjen izvedbi prilagoditev in izboljšav na PO. Pri tem lahko kot "čisto" vzdrževanje obravnavamo le odpravljanje napak, medtem ko ostali vidiki vzdrževanja sovpadajo z razvojem sistemov.

Slika 6: Povezava med procesom vzdrževanja in upravljanjem konfiguracij PO



Vir: Taramaa, 1998, str. 61

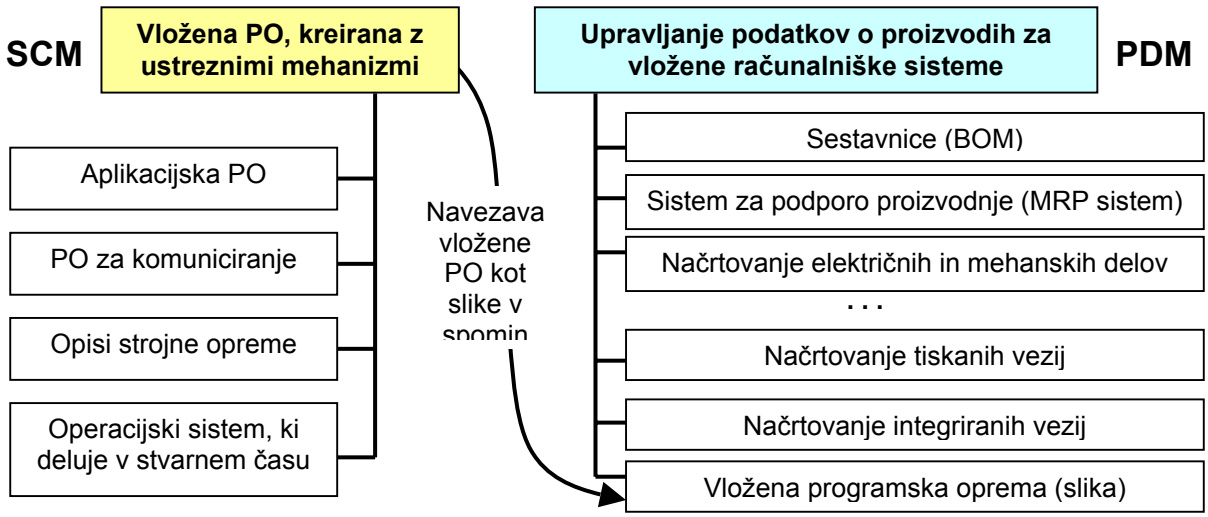
Pri vzdrževanju sistemov z vloženimi računalniškimi sistemi igra pomembno vlogo t.im. *tehnologija patchov*. Pod pojmom *patch* razumemo spremembo, izvedeno neposredno na objektnem programu, brez ponovnega prevajanja izvornega programa. V primerih, ko je sistem že nameščen na terenu, je uporaba patchov običajno najučinkovitejša metoda za spreminjanje njegove PO. Dinamični sistemi za ažuriranje programov omogočajo odpraviti obstoječe napake ali izboljšati PO, ne da bi bili potrebni dodatni stroški za zaustavljanje sistema (Taramaa, 1998, str. 52). Tovrstni prijemi se intenzivno uporabljajo tudi pri vzdrževanju PO telekomunikacijskih vozlišč SI2000.

2.1.6 Sistemi za upravljanje konfiguracij izdelkov in njihove programske opreme

Zadnji dve desetletji se v procesu upravljanja konfiguracij izdelkov vse bolj uporabljajo računalniška orodja. Njihov namen je ne le avtomatizirati proces upravljanja konfiguracij, temveč učinkovito podpreti upravljanje celotnega življenjskega cikla teh proizvodov.

Najbolj razširjen tip sistemov za podporo upravljanja konfiguracij izdelkov so t.im. PDM sistemi oziroma sistemi za upravljanje podatkov o proizvodih. Iz njih izhajajo tudi drugi sorodni tipi sistemov, npr. PLM sistemi, t.j. sistemi za upravljanje celotnega življenjskega cikla proizvodov, ki povezujejo naslednje tehnologije: integrirano načrtovanje in inženiring proizvodov (s povezanimi CAD/CAE sistemi), načrtovanje virov podjetja (angl. Enterprise Resource Planning, krat. ERP), upravljanje odnosov s kupci (angl. Customer Relationship Management, krat. CRM) in upravljanje z oskrbovalno verigo (angl. Supply Chain Management, krat. SCM). Drug pomembnejši in zelo poznan tip sistemov za upravljanje konfiguracij so SCM sistemi (sistemi za upravljanje konfiguracij PO). Lahko ugotovimo, da je temeljni namen obeh tipov sistemov enak: zagotoviti učinkovito upravljanje izdelka skozi njegov življenjski cikel. To dosežeta z učinkovitim upravljanjem konfiguracij in sprememb izdelka, ki sta temeljni funkciji obeh tipov podpornih sistemov. Bistvena razlika, ki ločuje oba tipa sistemov je ta, da so PDM sistemi namenjeni upravljanju industrijskih izdelkov, sestavljenih tako iz strojne kot iz programske opreme (lahko pa tudi iz drugih komponent), SCM sistemi pa so namenjeni upravljanju izdelkov, sestavljenih pretežno ali v celoti iz PO in z njo povezane dokumentacije. Medtem ko so si na ožjih področjih upravljanja konfiguracij (nadzor verzij/revizij gradnikov, nadzor konfiguracij, nadzor sprememb ipd.) funkcije PDM in SCM sistemov razmeroma podobne, SCM sistemi ponujajo dodatne funkcije za delo s programsko opremo, npr. na področjih integracije in gradnje PO. Sistemi upravljanja konfiguracije proizvodov so tesno vpeti v celotni življenjski cikel industrijskih izdelkov, za sisteme upravljanja konfiguracije PO pa to velja v nekoliko manjši meri. Le-ti namreč najbolj pridejo do izraza v fazah razvoja in vzdrževanja teh proizvodov, kjer pride do posebne veljave načrtovanje, implementacija in vzdrževanje PO, ki predstavlja ključno komponento vloženih računalniških sistemov. V teh fazah je zaželjeno, da sta oba omenjena sistema čim bolje povezana, brez nepotrebnega podvajanja operacij. V primeru upravljanja izdelkov z vloženimi računalniškimi sistemi povezuje PDM in SCM sistema vložena PO, ki se kot *slika* (angl. image) nalaga na vloženi računalniški sistem končnega proizvoda.

Slika 7: Povezava med SCM in PDM pri upravljanju proizvodov z vloženimi računalniškimi sistemi



Vir: Taramaa, 1998, str. 65

Poln izkoristek SCM, PDM, ERP in drugih sistemov za podporo različnih ravni poslovanja lahko dobimo šele z njihovo povezavo (integracijo) v celovit sistem. Pri tem ima vse večji pomen razumevanje celotnega procesa v podjetju, v katerem so v življenskem ciklu izdelkov udeleženi različni funkcionalni deli podjetja (oddelki za načrtovanje, finance, prodaja, proizvodnja itd.).
Vrzeli med SCM in PDM sistemi (Estublier, Favre, Morat, 1998, str. 90), prav tako pa tudi med PDM in ERP (Duhovnik, Tavčar, 2000, str. 1.13) ter drugimi sorodnimi tipi podpornih sistemov se zmanjšujejo. Analize množice primerkov PDM in SCM sistemov kažejo, da je v obeh tipih sistemov implementirana vrsta zelo sorodnih pristopov in tehnoloških rešitev (Westfechtel, Conradi, 1998, str. 103). Pričakuje se, da se bodo meje med tovrstnimi sistemi za podporo poslovanja v prihodnje še bolj zabrisale, ali z združitvijo sistemov za podporo poslovanja v celovitejše rešitve (primer sistemov mySAP.com in Teamcenter) ali z možnostmi učinkovitejšega povezovanja različnih sistemov s pomočjo povezovalne programske opreme (npr. aplikativni strežniki) ali standardi povezovanja (npr. XML). Pri tem igra pomembno vlogo Web tehnologija. Večina CM orodij že ima podporo za Web okolje, kar jim omogoča solidno integracijo z različnimi orodji oziroma aplikacijami (Burrows, 1999, str. 14).
V nadaljevanju so predstavljene osnovne lastnosti PDM in SCM sistemov.

2.1.6.1 Sistemi za upravljanje podatkov o proizvodih (PDM sistemi)

V dinamičnih poslovnih okoljih proizvodnih podjetij informacije o izdelkih tečejo v različnih smereh (CMstat, 2002): marketinški koncepti in informacije potujejo proti oddelkom za planiranje izdelkov in naprej proti skupinam za načrtovanje; skupine za načrtovanje in razvoj sodelujejo s skupinami v proizvodnji in servisno podporo; zatem informacije o problemih v zvezi z izdelki spet potujejo nazaj v oddelke za načrtovanje in izvajanje sprememb na izdelkih; informacije o naročilih in podatki o načrtovanju potujejo do dobaviteljev in drugih partnerjev; informacije o stroških potujejo v oddelke za finance, itd. V tem procesu je potrebno upoštevati vrsto zahtev in omejitev, npr.: izdelke je potrebno prilagoditi zahtevam različnih tržišč in različnim regulativam; zagotoviti je potrebno uporabo različnih (človeških, materialnih...) virov; uskladiti je potrebno lastni razvoj z razvojem, ki ga izvajajo pogodbeniki, ipd. Posledica tega je, da so postale tehnične informacije, potrebne za upravljanje izdelkov visoke tehnologije, izjemno zapletene. To po eni strani vpliva na upočasnitev procesa, po drugi strani pa na večje možnosti pojavljanja napak. Sledenje podatkov o izdelkih v teh okoliščinah zahteva ne le učinkovit proces upravljanja proizvodov in njihovih konfiguracij, temveč tudi zanesljiv podporni sistem, ki bo učinkovito podpiral delo z množico podatkov o proizvodih.
PDM sistem je orodje, ki podjetjem omogoča upravljanje (načrtovanje, spremljanje, sledenje...) inženirskih informacij o izdelkih, npr.: kje se določen del izdelka uporablja; kdo ga načrtuje; kako je načrtovan; kje, kako in kdaj se izdeluje, ipd. Uporablja se za delo z različnimi tipi podatkovnih datotek in zapisov. Ti lahko vključujejo: konfiguracije izdelkov, definicije izdelkov in njihovih sestavnih delov, specifikacije, modele in druge rezultate opravljenih inženirskih analiz, načrte proizvodnih in drugih procesov, avdio in video zapise, itd. Tovrstne informacije nastajajo skozi celotni življenjski cikel izdelka (Gascoigne, 1995). PDM sistemi zagotavljajo strukturo za njihovo učinkovito uporabo, pri čemer je na ustrezni ravni omogočena dvosmerna komunikacija med uporabnikom in sistemom. PDM sistemi morajo biti sposobni beležiti tako vsebino gradnikov kot posamezne atribute in njihove medsebojne povezave. V kratkem: v PDM sistemu se lahko vodi kakršna koli informacija, potrebna v toku življenjskega cikla izdelka, pri čemer je vseskozi dostopna vsakomur, ki jo potrebuje pri delu s proizvodom. Pri tem je pomembna, a preprosta ideja v zvezi z učinkovito uporabo PDM sistemov: "podatke vnesti enkrat in uporabiti večkrat" (Lyon, 2001, str. 109).

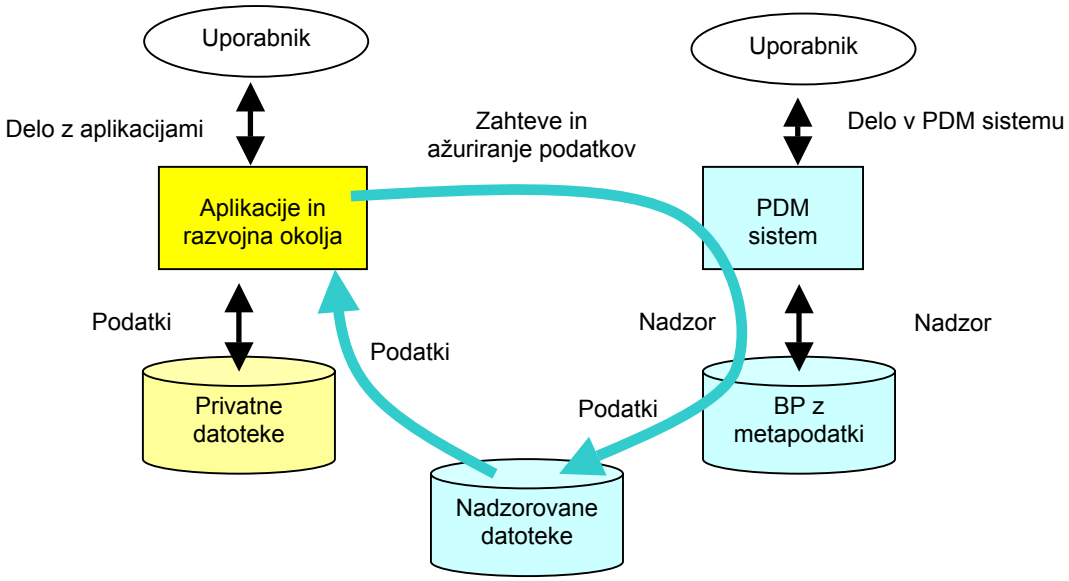
Upravljanje konfiguracij ni poseben modul, ki bi ga vgradili v PDM sistem, zato da bi ta bolje deloval, temveč je sestavni del PDM procesa (CMstat, 2002). Načrtovano je v samem jedru PDM sistema. Funkcije v zvezi z upravljanjem konfiguracij izdelkov so se v praksi pokazale kot tista ključna sposobnost PDM sistema, ki naredi vse druge funkcije PDM sistema uporabne. Tipični PDM sistem je sestavljen iz (CIMdata, 1999, str. 9):

- množice uporabniških funkcij;
- množice pomožnih funkcij;
- odlagališča (skladišča, "knjižnice"...) podatkov, ki običajno vključuje relacijsko podatkovno bazo.

Funkcije PDM sistemov, ki omogočajo, da se PDM sistemi uporabljajo v skladu z zahtevami, lahko razvrstimo takole (Gascoigne, 1995):

1. *Uporabniške funkcije:*
   - *upravljanje z izdajami rezultatov načrtovanja* - zagotavljajo varnost in nadzor nad doseganjem podatkov, povezave med podatki, mehanizme odjavljanja in prijavljanja gradnikov, upravljanje z metapodatki, liste uporabnikov ipd.;
   - *upravljanje strukture izdelkov* - zagotavlja funkcije za delo s kosovnicami in sestavnicami (angl. bill of materials, krat. BOM), povezave med gradniki izdelkov in zmožnost povezati načrt (opis) izdelka s strukturo in gradniki izdelka. Pomembna lastnost tega pristopa je v tem, da se pri tem ni potrebno omejevati le na fizično strukturo izdelka, temveč lahko vzpostavimo tudi druge vrste struktur, npr. proizvodno, finančno, vzdrževalno, dokumentacijsko (Hewlett-Packard, 2002). Tako lahko strokovnjak za dano področje vidi in upravlja izdelek z zornega kota, ki mu najbolje ustreza;
   - *razvrščanje dokumentacije in drugih komponent izdelka.* Ena od osnovnih funkcij PDM sistemov je razvrščanje informacij. Informacije podobnega tipa naj bi tako uredili v razrede, ustrezajoče določenemu tipu poslovnih zahtev. Podrobnejše razvrščanje naj bi dosegli z uporabo atributov za opis bistvenih lastnosti vsakega primerka (instance) danega razreda. Posamezne razrede lahko naknadno spet razvrščamo po skupinah. Ta način omogoča, da podjetje lahko ves svoj nabor komponent oziroma gradnikov razvrsti v pregledni hierarhični strukturi;
   - *nadzor nad procesi.* Poleg gradnikov izdelka se s PDM sistemom lahko nadzorujejo tudi procesi. V nasprotju z organiziranjem podatkov in informacij o izdelku, ki ima bolj opravka s statičnimi strukturami, se upravljanje procesa (Hewlett-Packard, 2002) ukvarja s tem, kako nadzorovati aktivnosti, s katerimi sodelujoči ustvarjajo in spreminjajo informacije o izdelkih. PDM sistemi običajno nadzorujejo: informiranje o izdelku, prehode med stanji izdelkov, procese v zvezi z obravnavo in potrjevanjem izdelkov in njihovih sestavnih delov, avtorizacijo itd. *Upravljanje procesa sprememb* podpira proces izvajanja inženirskih sprememb in proces obravnavanja in potrjevanja gradnikov konfiguracije. Vse to se običajno vrti okrog verzij gradnikov, njihovih povezav, nadzora nad njimi, izdelave izhodiščnih konfiguracij proizvodov ipd.
2. *Pomožne funkcije orodij (angl. utility functions):*
   - *komuniciranje* - zagotovlja komuniciranje znotraj PDM sistema in vmesnike z drugimi sistemi za komuniciranje (npr. elektronsko pošto);
   - *prenos podatkov* - zagotavlja mehanizme za prenos podatkov med uporabniki, aplikacijami, PDM funkcijami...;
   - *prevajanje podatkov* - zagotavlja doseganje orodij, ki prevajajo podatke med aplikacijami (npr. med CAD in CAM);
   - storitve glede *možnosti pregledovanja različnih (npr. grafičnih) objektov*;
   - *administriranje* - zagotavlja funkcije za postavitev, prilagoditev in upravljanje sistema.

Slika 8: Zgradba in funkcije PDM sistema



Vir: Product-data management, 1997

Odlagališče podatkov predstavlja osrednje, pooblaščeno mesto za hranjenje podatkov in metapodatkov o proizvodih. Osnovo odlagališča običajno predstavlja relacijska podatkovna baza. V njej so opisane povezave med kosovnicami in gradniki izdelka, procesi, dokumentacijo in spremembami teh elementov. Prav tako se tu hranijo atributi in drugi metapodatki o gradnikih in procesih. Odlagališče podatkov se poleg tega uporablja za shranjevanje in nadzor vsebine vseh gradnikov. Nadzira doseganje vse dokumentacije, ki se vodi v elektronski obliki, in datotek, proizvedenih z različnimi aplikacijami. To se lahko doseže ali z vgraditvijo PDM funkcij v druge aplikacije (npr. v CAD orodja) ali obratno. Za kakovostna odlagališča so v ustrezni meri izpolnjene naslednje administrativne zahteve (Marco, 2000, str. 87): varnost; hkratno doseganje; upravljanje sprememb; vrednotenje integritete in konsistentnosti podatkov in metapodatkov; možnost obnovitve ustreznega stanja v primeru napak.

Podatki, ki se vodijo v PDM sistemu, se običajno prenašajo naprej v druge sisteme za podporo poslovanja, npr. v proizvodni informacijski sistem (ERP, Enterprise Resource Planning), ki poleg podatkov o izdelkih podpirajo tudi (Duhovnik, Tavčar, 2000, str. 1.13) spremljanje nabave in prodaje, finančno poslovanje ipd. Ta pot je tako glede poteka poslovnega procesa, kot tudi glede zaporedja nastajanja informacij, najbolj naravna. Področji PDM in ERP sistemov se v marsičem prekrivata (npr. obvladovanje strukture proizvodov). Obstaja težnja po še večjem povezovanju in združevanju obeh tipov podpornih sistemov. V splošnem v praksi velja naslednje pravilo: čim bolj se poskuša oba sistema povezati, tem več podatkov se prenaša med PDM in ERP sistemom, zaradi česar se le-ti v določeni meri podvajajo - odločiti se je potrebno, kateri sistem je prevladujoč in kateri sistemi vsebuje izhodiščne podatke ter kako se le-ti prenašajo v drug sistem (Duhovnik, Tavčar, 2000, str. 1.14).

V kolikor PDM sistem razširimo, tako da poleg navedenih informacij o izdelkih vključuje še dokumentacijo vodenja kakovosti, vključno z notranjimi postopki, podatke o pogodbah, različne predloge ipd., tedaj govorimo o t.im. EDM (angl. Enterprise Data Management) sistemih, t.j. sistemih za upravljanje podatkov o poslovanju podjetja (Lyon, 2001, str. 109).

2.1.6.2 Sistemi za upravljanje konfiguracij programske opreme (SCM sistemi)

Podobno kot PDM sistemi je tudi večina SCM sistemov sestavljenih iz množice uporabniških funkcij, množice pomožnih funkcij in odlagališča (skladišča, "knjižnice"...) podatkov, ki tudi pri

SCM sistemih običajno vključuje relacijsko podatkovno bazo. Medtem ko se namen, v mnogih primerih pa tudi način izvedbe pomožnih funkcij in elektronskega odlagališča tako za PDM kot za SCM sisteme v veliki meri ujemata, pa se oba tipa sistemov najbolj značilno razlikujeta po množici uporabniških funkcij. Uporabniške funkcije za SCM sisteme:

- *nadzor verzij (angl. version control)* je temelj vsakega SCM sistema. Osnove nadzora verzij so vgrajene npr. v SCCS in RCS za Unix, ki prideta najbolje do izraza v majhnih projektih, lociranih na eni sami lokaciji. Ko projekti narastejo po velikosti in kompleksnosti, pa nadzor verzij zahteva več kot le osnove. Zahteva: upravljanje večkratnih vejitev objektov izvorne kode; učinkovito združevanje vejitev (merge); podporo dela na različnih geografskih lokacijah itd.;
- *upravljanje s konfiguracijami (angl. configuration management)* med drugim omogoča določiti, katere verzije komponent bodo vključene v neko izdajo paketa ali v določen projekt. S tem je zagotovljena sledljivost in ponovljivost v projektu;
- *nadzor nad nalogami (angl. issue management)* vsebuje beleženje in sledenje poročil o pomanjkljivostih in napakah. Nadzor nad nalogami vključuje še *obdelavo zahtev za spremembo* (angl. change requests);
- *upravljanje procesa (angl. process management)* omogoča vpeljavo in uporabo življenjskega cikla tako za gradnike (in komponente) z verzijami kot tudi za sam proces potrjevanja gradnikov. S tem je povezano *zagotavljanje varnosti, vezano na vloge* (angl. role-based security), ki jih imajo sodelujoči v projektu;
- *podpora delovnih okolij (angl. workspace management)* omogoča uporabnikom lasten pogled na projekt. Primer: razvijalcem je omogočeno spreminjanje in testiranje sprememb na PO, ne da bi pri tem vplivali na na delo ostalih razvijalcev;
- *upravljanje gradnje in izdajanja PO (angl. built management, release management).* Osnovna lastnost inteligentnih procesov za gradnjo PO je, da v največji možni meri ponovno uporabljajo rezultate prejšnjih gradenj. Podpora izdajanja PO omogoča razvijalcem sledenje, katere verzije komponent imajo uporabniki.

Kakovostno SCM orodje naj bi podpiralo vsa področja upravljanja konfiguracij za programsko opremo na uporabnikom prijazen način in ne da bi sililo poslovni sistem, da ta le zaradi omejitev orodja spreminja proces razvoja PO.

2.2 Kratek pregled pristopov k upravljanju s konfiguracijo izdelkov in njihove programske opreme

Funkcije upravljanja konfiguracij izdelkov in posameznih komponent so lahko vsebovane tako v posameznih aplikacijah, ki se uporabljajo za razvoj posameznih komponent, kot v posebnih sistemih za upravljanje konfiguracij izdelkov in njihovih komponent. Omenjene funkcije lahko predstavimo z enim ali večimi v nadaljevanju predstavljenimi modeli upravljanja konfiguracij.

Eno najbolj poglobljenih analiz osnovnih konceptov upravljanja konfiguracij je podal Feiler. Teorija ni najnovejša, je pa še vedno zelo aktualna. Njeno razumevanje predstavlja ključ do razumevanja tako starejših kot najsodobnejših CM sistemov. Poleg Feilerjeve teorije je predstavljen še Taramov pristop v zvezi z upravljanjem konfiguracij izdelkov, ki vsebujejo vložene računalniške sisteme.

2.2.1 Feilerjeva teorija modelov upravljanja konfiguracij

Na sisteme za podporo upravljanju konfiguracij ne moremo gledati neodvisno od procesa razvoja in vzdrževanja izdelkov (Feiler, 1991, str. 47). Podporni sistem vpliva na proces preko vgrajenih konceptualnih modelov upravljanja konfiguracij. Štiri obravnavane modele upravljanja konfiguracij lahko v kratkem orišemo takole (Feiler, 1991, str. 2):

- model odjava/prijava podpira upravljanje verzij komponent (pri predstavitvi te teorije je pojem 'komponenta' enakovreden pojmu 'gradnik') proizvoda;
- kompozicijski model se usmerja v izboljšave pri gradnji konfiguracije proizvoda, kar doseže z mehanizmi izbiranja ustreznih verzij gradnikov, nastopajočih v strukturi izdelka;
- s transakcijskim modelom lahko vodimo razvoj proizvodov kot zaporedje verzij njihovih konfiguracij, pri čemer je dan poudarek usklajevanju sočasno izvajajočih se razvojnih aktivnosti;
- model množice sprememb podpira upravljanje konfiguracij z vodenjem logičnih sprememb.

Običajno lahko v enem sistemu za podporo upravljanja konfiguracij izdelkov in njihove PO zasledimo po več omenjenih modelov. Največji izziv na tem področju je dobiti enoten model upravljanja konfiguracij, s katerim bi bili sposobni povezati različne obstoječe koncepte.

2.2.1.1 Model odjava/prijava

Model odjava/prijava je v osnovi namenjen podpori delu z verzijami posameznih datotek. Temeljna podsistema, ki nastopata v tem modelu, sta odlagališče podatkov in uporabnikov datotečni sistem.

2.2.1.1.1 Podpora delovnega okolja

Datotečni sistem predstavlja delovno okolje za uporabnika in njegova orodja. Odlagališče podatkov ne vsebuje in podpira mehanizmov, s katerimi bi bil zagotovljen nadzor nad tem datotečnim sistemom ali nad pravicami za spreminjanje vsebine odjavljenih verzij datotek. Uporabnik dela s svojimi orodji, vključno z orodjem za gradnjo, na enak način kot v drugih običajnih datotečnih sistemih. To pomeni, da ta orodja sama po sebi ne ločujejo različnih verzij gradnikov. Za strukturiranje gradnikov v celovit sistem se običajno uporablja hierarhična struktura datotečnih map.

2.2.1.1.2 Nadzor verzij in vejitev gradnikov

Za datoteke se vodijo verzije. Njihova vsebina se hrani v odlagališču podatkov.Datoteke se ne dosegajo neposredno. Uporabnik mora najprej izvesti operacijo *odjave* (angl. check out). S tem se ustrezna verzija dane datoteke prekopira v uporabnikov datotečni sistem. Pri tem obstaja več možnosti v zvezi s pravicami za izvajanje sprememb nad vsebino te datoteke:

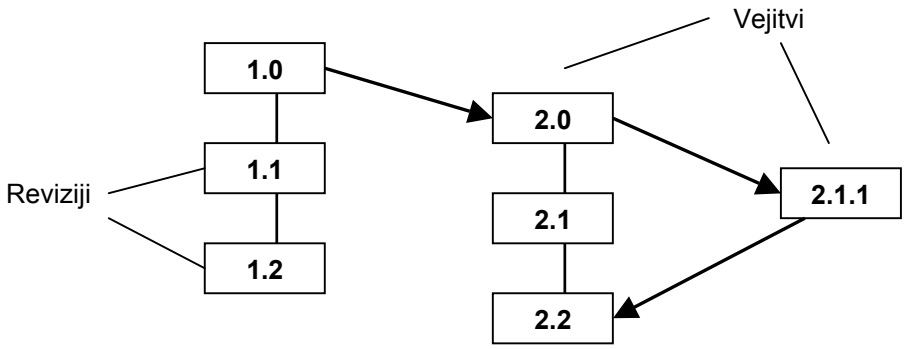
- ob odjavi je lahko vsebina verzije zaščitena pred pisanjem, dovoljeno je le branje (npr. za pregledovanje pravilnosti vsebine verzije gradnika);
- ob odjavi ima uporabnik pravico do spreminjanja. V tem primeru morajo biti zagotovljeni dodatni mehanizmi, ki nadzirajo večkratne tovrstne odjave.

Ko uporabnik spremeni vsebino odjavljene verzije gradnika, jo vključi oziroma *prijavi* (angl. check in) nazaj v odlagališče. Kreira se nova verzija te datoteke.

Model podpira nastajanje zaporednih verzij gradnika izdelka, prav tako pa tudi vejitve (angl. branches). Slednje nastanejo iz določene verzije v obstoječi veji, pri čemer iz te verzije nastaneta na neodvisen način dve novi verziji. Poleg tega model podpira združevanje verzij iz različnih vejitev v novo verzijo, ki spada v eno od obstoječih vejitev. Rezultat teh operacij je

zgodovina verzij za dano datoteko. Običajno jo predstavljamo v grafični obliki, zato se sklicujemo nanjo kot na *graf verzij* (glej sliko 9).

Slika 9: Graf verzij - vejitve in združevanje verzij



Vir: Feiler, 1991, str. 7

Vejitve lahko uporabljamo z različnimi nameni, kot npr.: za predstavitev različnih razvojnih poti; za predstavitev eksperimentalnega razvoja, ki ga pozneje lahko opustimo; za izvajanje popravkov na starih verzijah gradnikov; za podporo sočasnega izvajanja različnih sprememb na isti verziji gradnika, itd.

2.2.1.1.3 Nadzor sočasnega spreminjanja vsebine verzij gradnikov

Nadzirajo se odjave tistih verzij gradnikov, ki jih uporabniki lahko spreminjajo. Zadnja verzija v dani veji se odjavi s podeljeno pravico spreminjanja, pri čemer se vejitev zaklene. Zaklepanje vejitve zagotavlja, da le ena oseba ob danem času lahko ustvari novo verzijo v določeni vejitvi. Ko se nova, spremenjena verzija datoteke doda v vejo, se vejitev zopet sprosti oziroma odklene. Ko je odjavljena verzija datoteke prenešena v datotečni sistem, je izven nadzora sistema za upravljanje konfiguracij. Običajno pa sistem lahko nadzoruje doseganje verzij datotek v odlagališču podatkov. Lahko se npr. vodijo seznami uporabnikov, ki imajo pravico spreminjati verzije dane datoteke. Omejitev odjavljanja na eno osebo z dane liste oziroma na eno odjavo v kateri koli časovni točki ima za posledico učinek zaklepanja.

2.2.1.1.4 Nadzor strukture sistema

Struktura proizvoda se lahko odraža v organizaciji datotečnih map v odlagališču podatkov. Prvi nivo poddirektorijev ponazarja podsisteme proizvoda, drugi nivo poddirektorijev pa komponente podsistema. Pri CM sistemih, ki tovrstne organiziranosti sistemskih datotek ne omogočajo, se lahko na vsaki datotečni mapi, ki ponazarja del strukture proizvoda, ustvari lastno odlagališče podatkov.

Skripti, ki se uporabljajo pri gradnji komponent sistema, nosijo običajno v sebi dodatno informacijo o strukturi sistema. To so npr. odvisnosti med datotekami oziroma gradniki izdelka, ki so lahko vključene že v vsebine verzij teh gradnikov. Ti skripti, ki jih običajno vzdržujejo razvijalci, se lahko obravnavajo kot običajni gradniki izdelka: shranjujejo se v odlagališču podatkov, zanje se vodijo verzije.

Določena odlagališča podatkov podpirajo označevanje posameznih verzij ali vejitev z oznakami (angl. labels). V takem primeru je informacija, katera verzija gradnika je vključena v dano konfiguracijo izdelka, vključena v drevo verzij. Velja naslednje pravilo: verzija gradnika, ki pripada dani konfiguraciji, je označena z oznako, ki določa dano konfiguracijo. V tovrstnih sistemih se torej vodijo dodatni objekti, ki določajo različne konfiguracije proizvoda.

S pravkar omenjenimi oznakami je že možno do določene mere improvizirati *logične spremembe* oziroma skupine, množice gradnikov, ki pripadajo dani spremembi ali zahtevi za

spremembo. Koncept vodenja logičnih sprememb je zelo aktualen pri upravljanju družine proizvodov. Spremenjene verzije gradnikov preprosto označimo z oznako, ki zaznamuje tisto zahtevo za spremembo izdelka, ki je vzrok za njihov nastanek. Pri tem pa je v praksi potrebno odpraviti vrsto omejitev, predvsem pri postavljanju mehanizma, ki bo sposoben tovrstne logične spremembe povezati brez protislovij in nekonsistentnosti v novo konfiguracijo proizvoda. Pri CM sistemih, ki temeljijo le na modelu odjava/prijava (to sta npr. RCS ali Unixov SCCS), je to razmeroma težka naloga.

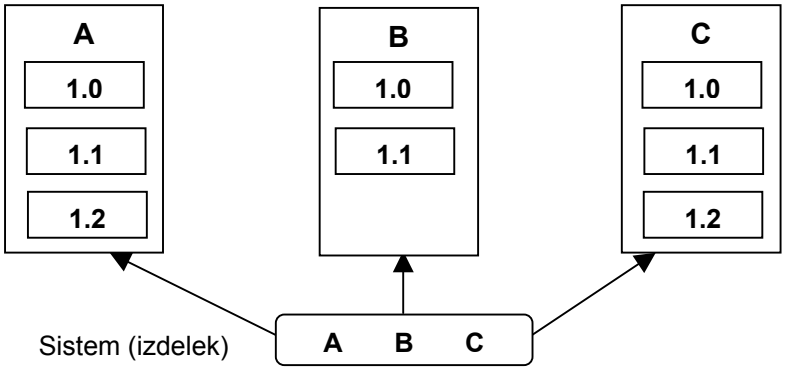
2.2.1.2 Kompozicijski model

Kompozicijski model oziroma model sestavljanja temelji na uporabi odlagališča podatkov, delovnega okolja in nadzoru sočasnega načina dela ob uporabi mehanizma zaklepanja. Ta model izhaja iz modela odjava/prijava in tako kot ta še vedno podpira vodenje verzij za posamezne komponente izdelka. Tudi tu se uporablja graf verzij.

2.2.1.2.1 Obravnava konfiguracij

Ta model omogoča obravnavati konfiguracije izdelka kot posebne entitete. Z njimi si razvijalci pomagajo pri oblikovanju delovnih okolij. Razvijalec dela s konfiguracijami v takšnem sistemu tako, da v prvem koraku postavi strukturo sistema oziroma sestavi sistem iz njegovih komponent, nato pa za vsako komponento izbere ustrezno verzijo (Feiler, 1991, str. 16).

Slika 10: Verzije gradnikov, komponent sistema



Vir: Feiler, 1991, str.15

Konfiguracijo izdelka v tem modelu sestavljata *model sistema* (proizvoda) in *pravila za izbiro verzij gradnikov*. Model sistema in pravila izbire se lahko obravnavata kot običajni komponenti in ju lahko v obliki verzij shranjujemo v odlagališču podatkov. Vsak od njiju nosi le del informacij v zvezi z določanjem konfiguracij. Šele ko nastopita skupaj, določata neko instanco konfiguracije (Feiler, 1991, str. 20). Model sistema vsebuje vse komponente oziroma gradnike, ki sestavljajo proizvod. Pravila za izbiro verzij določajo, katero verzijo izbrati za vsak gradnik, da dobimo dano konfiguracijo. S pravili za izbiro verzij tako *omejimo* (angl. bind) gradnik na določeno verzijo.
Opisana dva koraka določanja konfiguracije (sestavljanje in izbira) lahko modeliramo z IN/ALI grafi. Prvi korak je sestavljanje komponent izdelka (IN-vozel). Naslednji korak je izbira ustrezne verzije za vsako od nastopajočih komponent (ALI-vozel).

2.2.1.2.2 Modeli sistemov in konsistentnost konfiguracij

Model sistema predstavlja strukturo proizvoda in vsebuje seznam vseh njegovih komponent. Te se lahko grupirajo, skupki komponent pa so lahko del drugih skupin komponent. Tako nastane hierarhična struktura. To lahko razširimo z odvisnostmi, tako z *nameravanimi* (vključevanje datotek) kot *dejanskimi* (klici procedur). Informacija, ki jo vsebujejo modeli sistema, je lahko vključena na različnih krajih. Struktura sistema je lahko predstavljena s hierarhijo datotečnih

map, lahko pa npr. tudi s hierarhijo konfiguracije v odlagališču podatkov. Skripti, ki se uporabljajo pri gradnji, običajno vsebujejo tako informacijo o strukturi sistema kot o odvisnostih. Tudi same komponente sistema lahko vsebujejo informacijo o strukturi in odvisnostih, npr. v obliki jezikovnih konstruktov.
Ena glavnih funkcij podpore upravljanja konfiguracij na tem nivoju je v izbiri verzij komponent na tak način, da bodo le-te tvorile konsistentno konfiguracijo. Konfiguracija je *konsistentna z ozirom na verzije*, če je verzija kakršne koli komponente, nastopajoče v proizvodu, usklajena z vsemi drugimi nastopajočimi verzijami. Naloga sistema za gradnjo je vzdrževanje ažurnosti izpeljanih (deriviranih) komponent

2.2.1.2.3 Pravila za izbiro verzij

Uporabniku ni potrebno ročno izbirati verzij komponent, ki jih želi iz odlagališča podatkov dobiti v svoje delovno okolje. Pravilo izbire lahko enolično določa verzijo vsake nastopajoče komponente. V tem primeru pravimo, da je konfiguracija, opisana z modelom sistema in pravili izbire, *omejena*. V nasprotnem primeru pa je konfiguracija *delno omejena* - govorimo o *vzorcu konfiguracije* (angl. configuration template). V takem primeru uporaba pravila izbire lahko v različnih časovnih točkah rezultira v različnih omejenih konfiguracijah - npr. ko pravilo izbire določa zadnjo verzijo komponente. Pravila izbire lahko opišemo na dva načina:

- *iskalne poti.* Na pravila izbire lahko gledamo kot na iskalne poti v grafu verzij z označenimi verzijami in vejitvami. V takšni iskalni poti uporabnik označi zaporedje možnosti izbiranja, v skladu s katerimi bo izbrana verzija katere koli od nastopajočih komponent. Možnosti izbire lahko vključujejo: delovna področja uporabnikov; označene vejitve, ki predstavljajo različne razvojne tokove ipd. Možnosti izbire se lahko nanašajo na vse komponente ali le na izbrane podmnožice. Razne variante proizvoda lahko dobimo s spremembo ene ali večih nastopajočih možnosti izbire;
- *uporaba objektov.* Drug način predstavitve pravil izbire temelji na uporabi objektov in njihovih atributov. Komponente izdelka, prav tako pa njihove verzije, se obravnavajo kot objekti. Atributi so npr.: identifikator verzije; datum spremembe; stanje objekta itd. Pravila izbire so predikati izjavnega računa prvega reda. Verzija komponente, ki ustreza predikatu, predstavlja del izbrane konfiguracije. Primer predikata: WindowSystem = X11 *and* WSRelease > R3 & (status = tested *or* reserved = me).

V praksi se je pokazalo, da je druga rešitev splošnejša od prve, vendar jo je praviloma težje implementirati.

2.2.1.2.4 Konfiguracije kot delovno okolje in način razvijalčevega dela

Razvijalci lahko izberejo določeno konfiguracijo kot svoje delovno okolje *(delovni kontekst)*. To pomeni, da se pri tem, ko razvijalec dosega različne komponente sistema, za vsako komponento najprej izbere tista verzija, ki ustreza pravilom izbire. Pri tem se lahko verzije komponent dosegajo posredno, preko orodij in prevzemanja verzij iz podatkovnih odlagališč, ali na neposreden način, transparentno.
Omejene konfiguracije predstavljajo stabilnejše delovno okolje razvijalca in pridejo v poštev npr. takrat, ko prevladuje zahteva po izoliranem delu razvijalca. Delno omejene konfiguracije imajo tudi svoje prednosti. Lahko se uporabljajo npr. tedaj, ko hoče razvijalec v svoje delovno okolje vključevati tudi zadnje oziroma aktualne spremembe drugih članov projektne skupine.
Konfiguracija, določena z modelom sistema in pravili izbire, predstavlja razvijalčevo začetno delovno okolje za izvajanje spremembe. Razvijalec nato spremeni pravila izbire v toliko, da se v njegovo delovno okolje vključijo tudi odjavljene komponente. Ko razvijalec zaključi delo, shrani novo konfiguracijo v odlagališče podatkov s tem, da prijavi vanj novo verzijo izdelka. To

zagotovi z vključevanjem vseh spremenjenih verzij komponent izdelka. Sledi še določanje novih pravil izbire, ki določajo novo stanje konfiguracije.

2.2.1.2.5 Upravljanje logičnih sprememb in podpora družinam proizvodov

Sistemi upravljanja konfiguracij, ki podpirajo ta model, obravnavajo spremembe kot logične celote. Model predpostavlja, da razvijalec izvede spremembo na neki množici gradnikov v svojem delovnem okolju. Kot dodatek k delovnemu okolju se privzame oznaka zahteve za dano spremembo. Ko uporabnik na kakršen koli način vključi novo, spremenjeno verzijo komponente v odlagališče podatkov in ji dodeli omenjeno oznako, podporni sistem razpozna novo verzijo komponente kot del logične spremembe, določene s to oznako. To omogoča enostavnejše obravnavanje verzij, ki predstavljajo spremembo.
Ta podpora spremembam ima tudi svoje omejitve. Ni npr. mogoče na enostaven način ugotoviti, ali je določena sprememba vsebovana v neki konfiguraciji.

2.2.1.3 Transakcijski model

Transakcijski model, imenovan tudi model dolgih transakcij, se usmerja na podporo razvoju celotnih sistemov (proizvodov) kot zaporedja posameznih sprememb in na usklajevanje sprememb na sistemu, ki jih izvajajo skupine razvijalcev. *Dolga transakcija* se razlikuje od običajne transakcije nad podatkovno bazo po tem, da slednja izvede operacijo ažuriranja baze podatkov, medtem ko dolga transakcija ustvari novo verzijo spremenjenega podatkovnega elementa. Ena dolga transakcija se lahko izvaja zelo dolgo - ure, dneve, mesece... Med njenim izvajanjem se lahko npr. zamenja uporabljena računalniška infrastruktura.

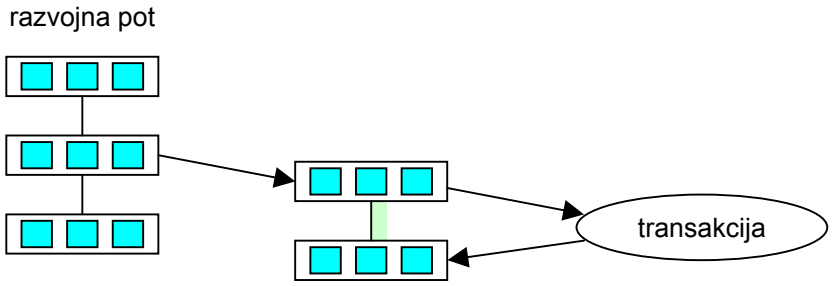
2.2.1.3.1 Delo s konfiguracijami

V transakcijskem modelu razvijalci delajo z verzijami konfiguracij. Najprej izberejo verzijo konfiguracije izdelka, nato se usmerijo na strukturo izdelka. Verzija komponente je določena posredno, z verzijo konfiguracije. V tem se ta model razlikuje od kompozicijskega modela, ki uporabnika najprej usmeri v strukturo sistema, nato pa v izbiro verzij komponent sistema. Transakcijski model sestavljata dva glavna koncepta:

- *delovni prostor* oziroma delovno okolje se navezuje na uporabo lokalnega sistema. Podporni sistem kot delovno področje ne uporablja več datotečnega sistema in ima nadzor tudi nad razvijalčevim delovnim okoljem;
- *shemo za nadzor sočasnih dogodkov* sestavljajo pravila, namenjena usklajevanju sočasno se izvajajočih sprememb. Ta pravila omogočajno različne stopnje sodelovanja in sočasnega dela v skupini razvijalcev. Tipični primer takšnih pravil so razni protokoli v bazah podatkov, s katerimi se na ustrezen način organizirajo transakcije, odvijajoče se v določenem obdobju.

Ob začetku izvajanja spremembe se izbere ustrezna omejena konfiguracija. Spremembe na gradnikih niso vidne izven ustrezne transakcije, dokler le-ta ni potrjena. Večkratne transakcije se usklajujejo s pomočjo shem za nadzor hkratnih dogodkov. S tem je zagotovljeno, da se izvedene spremembe ne izgubljajo. Potrditev transakcije ima za posledico novo verzijo konfiguracije. Rezultat zaporedja izvedenih sprememb je zaporedje verzij konfiguracij, imenovano tudi *razvojna pot*. Verzije dane konfiguracije se lahko vejijo glede na obstoječo razvojno pot. S tem nastane nova, *neodvisna razvojna pot*. Tako se imenuje zato, ker se običajno obe takšni poti razvijata neodvisno druga od druge (glej sliko 11).

Slika 11: Zgodovina verzij dane konfiguracije

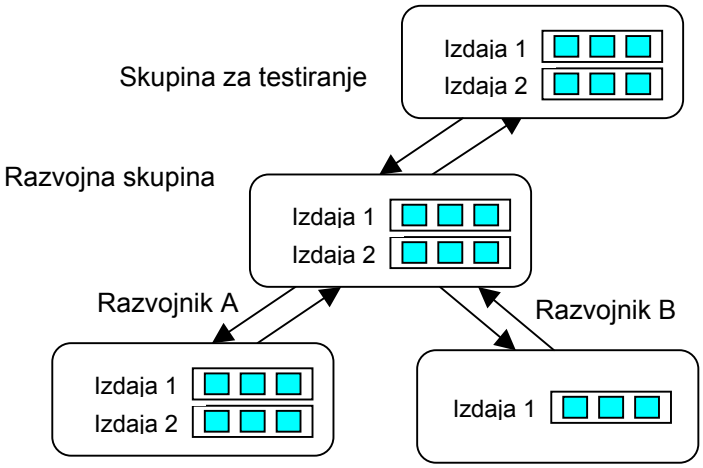


Vir: Feiler, 1991, str.23

2.2.1.3.2 Upravljanje delovnih prostorov

Delovni prostor razvijalca je tako sestavljen iz delovne konfiguracije in zaporedja ohranjenih konfiguracij. Delovna konfiguracija predstavlja konfiguracijo, v kateri se lahko aktivno spreminjajo komponente izdelka. Zgodovina sprememb konfiguracij se vodi na lokalnem sistemu v obliki nekomutativnega kompozituma. Pri tem je potreben mehanizem, ki vse spremembe komponent prevede v spremembo celotne konfiguracije. Zaporedje ohranjenih konfiguracij razvijalcu nudi kontrolne točke, na katere se pozneje lahko vrača.

Slika 12: Transakcije kot delovne naloge



Vir: Feiler, 1991, str.27

Medtem ko razvijalec dela v svojem delovnem okolju, je izoliran od sprememb drugih razvijalcev. Tako operacija ustvarjanja oziroma spreminjanja delovnega okolja kot operacija promoviranja izvedenih sprememb sta eksplicitno pod nadzorom uporabnika. Vse spremembe v delovnem prostoru se vodijo z ozirom na izhodiščno konfiguracijo. Pri tem je dovoljeno gnezdenje delovnih prostorov. Namesto, da bi začetno konfiguracijo dobili iz odlagališča podatkov, jo lahko ustvarimo iz shranjene konfiguracije drugega delovnega prostora. To ima za posledico hierarhijo delovnih prostorov, določeno s hierarhijo gnezdenih transakcij (gl. sliko 12). Podporni sistem, ki ustreza transakcijskemu modelu, podpira transparentno doseganje odlagališča podatkov preko delovnih prostorov. Ko razvijalec vstopi v delovno okolje, takoj začne delati na strukturi izdelka. Kopiranje komponent v datotečni sistem delovnega okolja ni potrebno. Pri vstopanju v delovno okolje sta najpogostejši naslednji možnosti:

- *datotečne mape.* Delovni prostori in njihove konfiguracije se pojavljajo kot datotečne mape. Izbira delovnega prostora in ustrezne verzije konfiguracije je vključena v pot do dane delovne datotečne mape. Dani proces lahko npr. dosega različne konfiguracije preko izbire dveh različnih poti datotečnih map;

- *uporaba mehanizma mount-ov.* Pri tej možnosti razvijalec pripravi podporni sistem do tega, da ta preslika določeno konfiguracijo na določeno točko v datotečnem sistemu. Doseganje do konfiguracij preko datotečnih map in datotek, ki so dejansko shranjeni v odlagališču podatkov, ki ga upravlja podporni sistem, se izvaja na podoben način, kot ga zagotavlja mehanizem t.im. *mount-ov* v sistemu NFS (Network File System). Vendar v nasprotju z NFS-mounti omenjeni mount obstaja le, dokler traja sam proces, ki ta mount uporablja, in njegovi potomci.

Nadzor delovnih prostorov omogoča podpornim sistemom, da vpeljejo in izkoristijo različne tehnike za optimizacijo porabe prostora, tako v odlagališčih podatkov kot v lokalnih delovnih prostorih. Primer takšne optimizacije je uporaba t.im. *virtualnih kopij konfiguracij* - na lokalnem sistemu se nahajajo le fizične kopije spremenjenih gradnikov, ostali gradniki pa se dosegajo neposredno iz odlagališča podatkov.

Podporni sistem za vsako konfiguracijo v delovnem okolju nadzoruje tudi izpeljane objekte. Ko se konfiguracija shrani, se hkrati shranijo tako izvorne komponente kot tudi iz njih izpeljane komponente. Pri zagotavljanju konsistentnosti izpeljanih objektov obstajata vsaj naslednji dve možnosti: (a) podporni sistem zagotovi, da gradnja oziroma ustvarjanje izpeljanih komponent uspe, še predno shranimo konfiguracijo; (b) podporni sistem samodejno poskrbi za brisanje neažurnih izpeljanih komponent.

2.2.1.3.3 Podpora sočasnemu razvoju

Podporo sočasnemu razvoju lahko razvrstimo v tri kategorije: (a) sočasno delo v enem delovnem prostoru; (b) sočasno in usklajeno delo v večih delovnih prostorih; (c) sočasno in medsebojno neodvisno delo. Prva dva načina se nanašata na isto konfiguracijo, medtem ko se pri tretjem načinu predpostavlja, da sistem nastaja preko neodvisnih razvojnih poti, med katerimi ni potrebno sproti zagotavljati usklajenosti. Ta se zagotovi pozneje oziroma kadar je to potrebno, s pomočjo mehanizma prenašanja sprememb v ustrezne delovne tokove. Pri tem se uporabi operacija združevanja vsebine ustreznih verzij gradnikov iz različnih razvojnih poti. Usklajevanje sočasnega razvoja se pojavlja, ko uporabniki delajo v različnih stabilnih delovnih okoljih, pri čemer bodo njihove spremembe vključene v novo verzijo izdelka. Sheme za nadzor sočasnega dela lahko razdelimo v dva razreda:

- *konservativne sheme*, ki izvajajo vnaprejšnja zaklepanja objektov;
- *optimistične sheme*, ki dovoljujejo hkratno izvajanje sprememb, konfliktne situacije pa se rešujejo pozneje, npr. pri prijavljanju sprememb v odlagališča podatkov.

2.2.1.3.4 Promoviranje sprememb in podpora razvoju in vzdrževanju družin proizvodov

Tako verzije konfiguracij kot verzije sprememb so lahko v enem od stanj ('v razvoju', 'testirano', 'izdano' ipd.). Stanje odraža dejstvo, da verzija ustreza določenim pogojem. Nabor stanj verzij ustreza procesom oziroma življenjskim ciklom, ki jih za izbrane objekte podpira sistem upravljanja konfiguracij. Sistemi, ki podpirajo transakcijski model, omogočajo ustvarjanje delovnih prostorov, posvečenih določenemu stanju konfiguracije oziroma spremembe. Ko je npr. konfiguracija v takšnem delovnem prostoru potrjena, se njeno stanje spremeni, nadzor nad njo pa se prenese v delovni prostor, ki je v hierarhiji gnezdenja za en nivo višje. Verzije konfiguracij s promoviranjem svojih stanj torej napredujejo po hierarhiji delovnih prostorov. Takšen pristop ima poleg prednosti tudi slabosti, npr. v slabši preglednosti nad stanjem dane konfiguracije v določeni časovni točki, saj delovna okolja niso neodvisna od trenutnih stanj konfiguracije.

Model podpira razvoj in vzdrževanje družin proizvodov na dva načina:

- zaporedne verzije proizvoda se hranijo v delovnem prostoru. Predstavljene so z razvojno potjo. Različice proizvoda so predstavljene z različnimi potomci delovnega prostora, ki omogočajo neomejen razvoj različic. Spremembe, izvedene na eni različici, se prenašajo na druge različice v okviru nadrejenega, skupnega delovnega prostora. Različice se izbirajo s pomočjo njihovih razvojnih poti;
- nekateri sistemi podpirajo delo z različicami znotraj enega delovnega prostora. Podpirajo npr. različne izpeljane gradnike za dano konfiguracijo, sestavljeno iz izvornih gradnikov. To uporabnika vodi k temu, da razvije vse različice drugo za drugo, predno potrdi oziroma shrani novo verzijo konfiguracije.

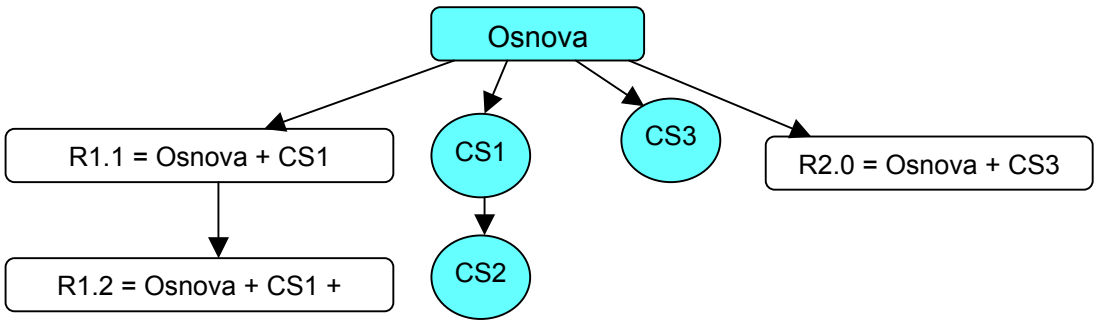
2.2.1.4 Model množic sprememb

Model množic sprememb oziroma natančneje, model množic, ki predstavljajo spremembe, se usmerja na podporo upravljanja logičnih sprememb konfiguracij proizvoda. Zapis o logični spremembi obstaja tudi potem, ko je sama aktivnost spreminjanja končana. Ta model upravljanja konfiguracij, ki ga nekateri imenujejo tudi *v spremembe usmerjeno upravljanje konfiguracij*, se razlikuje od t.im. *v verzije usmerjenega upravljanja konfiguracij*, ki je značilen za prejšnje tri modele in se usmerja v upravljanje verzij konfiguracij ter njihovih komponent.

2.2.1.4.1 Povezava med konfiguracijami in množicami sprememb

Koncept množice sprememb predstavlja naravno povezavo do zahtev za spremembo (angl. change requests). Zahteve za spremembo so orodje upravljanja s spremembami. Predstavljajo zapis o stanju spremembe v procesu spreminjanja izdelkov. Medtem ko zahteve za spremembe vsebujejo informacije o spremembah, množice sprememb predstavljajo dejanske spremembe. V kontekstu posamezne komponente je množica sprememb *množica razlik* (t.im. *delt*) med dvema verzijama datotek. Z ozirom na celotno konfiguracijo množica sprememb predstavlja množico razlik med dvema verzijama konfiguracije. Ta množica razlik je zbirka delt tistih komponent, ki so bile spremenjene med dvema verzijama konfiguracije - to so pravzaprav delte elementov iz množice spremenjenih komponent. V nasprotju s prejšnjimi modeli lahko tu razvijalec poimenuje spremembe in jih obravnava kot entitete. Pri tem se lahko sklicuje tudi na množico množic, ki predstavljajo spremembe, ne le na posamezne spremembe.
V tem modelu so konfiguracije določene z izhodiščno konfiguracijo in množico, katere elementi so množice sprememb. Konfiguracije lahko opišemo na dva načina: tako z grafom verzij konfiguracij (pravokotniki na sliki 13) kot z grafom verzij množic sprememb (obarvane elipse na sliki 13). Dvojnost opisa temelji na dejstvu, da verzijo konfiguracije lahko opišemo z izhodiščno konfiguracijo in množicami izvedenih sprememb.

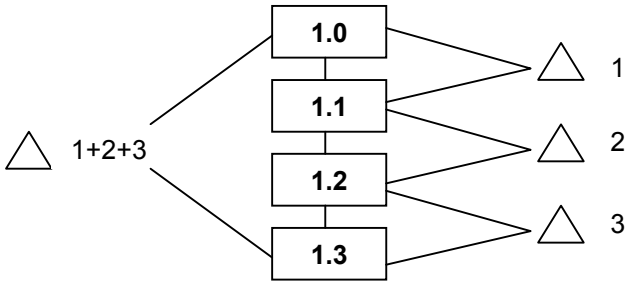
Slika 13: Graf verzij konfiguracij in ustrezne množice sprememb



Vir: Feiler, 1991, str.40

Razvijalci lahko sledijo logičnim spremembam in lahko ugotavljajo, ali so te spremembe del dane konfiguracije. Lastnost, da z množicami sprememb delamo kot z entitetami, med drugim dovoljuje razvijalcu izvajati naslednje poizvedbe po informacijah v sistemu: (a) Katere komponente so se spremenile kot del logične spremembe? (b) Katere množice sprememb so vključene v dano konfiguracijo? (c) V katere konfiguracije je vključena dana sprememba?

Slika 14: Kumulativne spremembe



Vir: Feiler, 1991, str.39

Verzija konfiguracije pravzaprav vsebuje zbirko zaporednih sprememb oziroma delt v grafu verzij (glej sliko 14). Iz tega sledi, da ugotavljanje, ali je določena sprememba del neke konfiguracije, ni le preprosta poizvedba po oznakah posameznih verzij, temveč zahteva pregledovanje grafa verzij.

2.2.1.4.2 Prenos sprememb na druge konfiguracije in ohranjanje konsistentnosti

Spremembe se lahko prenašajo na druge konfiguracije z uporabo ustreznih množic sprememb. Pri vključevanju zbirke logičnih sprememb v novo izdajo se lahko uporabljajo različne integracijske strategije. Prenašanje sprememb med konfiguracijami se navadno izvaja s pomočjo združevanja verzij. Ko se neka verzija združi z drugo, se prenese kumulativna množica sprememb. Pri prenašanju sprememb se lahko pojavijo težave, npr. v primerih, ko ustrezni podporni sistem vključuje lastnosti in s tem omejitve katerega od prejšnjih treh modelov (primer: omejitve za hierarhijo transakcij v transakcijskem modelu).

Model množice sprememb kot rečeno podpira izdelavo nove konfiguracije z dodajanjem množic sprememb izhodiščni konfiguraciji. Pri tem pa običajno obstajata naslednja tipa omejitev v zvezi z ohranitvijo konsistentnosti: določene množice sprememb so lahko *odvisne* od drugih ali pa so *v konfliktu* z drugimi množicami sprememb. Konflikti se lahko pojavljajo v naslednjih primerih:

- pri prenašanju logične spremembe v drugo obstoječo konfiguracijo, npr. ob združevanju spremembe iz ene razvojne poti v drugo, ali
- ob povezovanju (integraciji) množice logičnih sprememb v eno izdajo izdelka.

Tako pri ugotavljanju odvisnosti kot pri ugotavljanju konfliktov med dvema množicama sprememb potrebujemo informacijo o logični odvisnosti oziroma neodvisnosti danih množic sprememb. Pri tem je potrebno preučiti tako *množico zapisanih sprememb* (t.s. spremembe v vsebini komponent), kot tudi *množico prebranih sprememb*, t.j. tistih lastnosti komponent, ki se v opravljenih spremembah tako ali drugače uporabijo.

2.2.1.4.3 Podpora družinam izdelkov in sočasnim spremembam v distribuiranem sistemu

Različice družine izdelkov so navadno predstavljene z vejitvami v grafu verzij konfiguracij. Model podpira mehanizme za prenašanje izvedenih sprememb na ostale različice družine. Spremembe se običajno izvedejo na eni različici proizvoda v eni razvojni poti in so zapisane v množici sprememb ene vejitve konfiguracije. Te spremembe nato lahko prenašamo na druge različice proizvodov oziroma na druge razvojne poti.

Model množic sprememb sam po sebi ne zagotavlja mehanizma zaklepanja, s katerim bi nadzoroval sočasno delo. V ta namen podporni sistem, ki vključuje model množice sprememb, običajno uporabi model odjave/prijave. Vendar pa lahko tudi z modelom množice sprememb dosežemo določeno raven nadzora nad hkratnim delom. Lahko npr. postavimo omejitve na prenašanje sočasno se izvajajočih logičnih sprememb, ki imajo v svojem grafu verzij vejitve.
Množice sprememb se lahko uporabljajo tudi pri necentraliziranem nadzoru sočasnega spreminjanja izdelkov v geografsko porazdeljenem razvojnem sistemu. Vsaka razvojna lokacija neodvisno proizvaja množice sprememb. Potem, ko se množice sprememb izmenjajo med lokacijami, lahko vsaka od njih vključi te spremembe, kot ji ustreza, v svoje komponente. Posledica tega je, da sistem nastaja neodvisno na večih razvojnih lokacijah. Če se dodeljevanje izvajanja sprememb različnim lokacijam planira na ustrezen način, se pri tem pojavlja razmeroma majhno število konfliktnih situacij.

2.2.1.5 Uporabnost naštetih pristopov

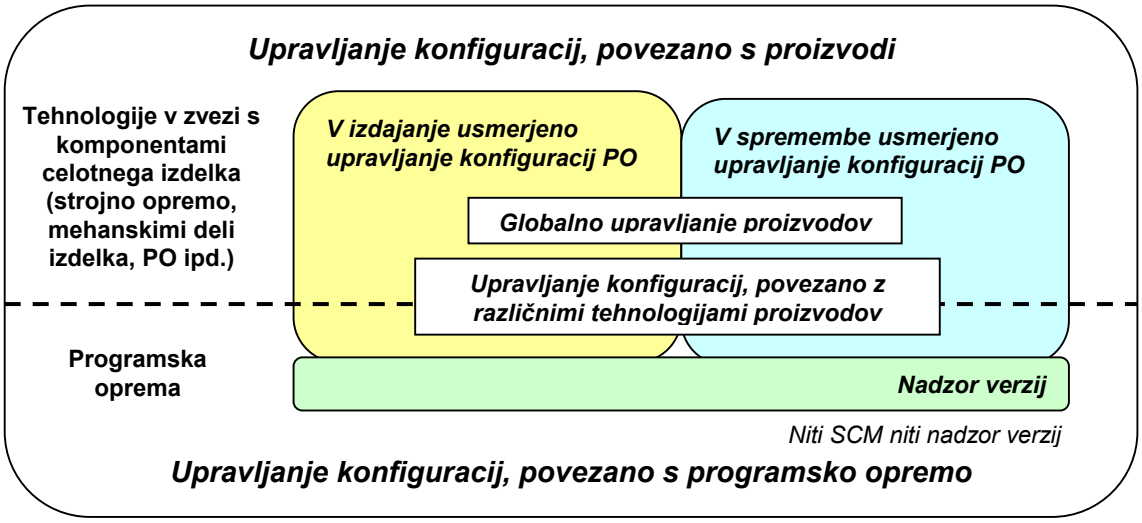
Domala vsi najbolj znani sistemi za upravljanje konfiguracij "razumejo" in tudi vključujejo model odjava/prijava. To je model, ki predstavlja enostavno in učinkovito osnovo za upravljanje sprememb tako na ravni posameznih elementarnih, atomarnih gradnikov kot na ravni programskih paketov ali celotnih proizvodov.
PDM sistemi večinoma uporabljajo model odjava/prijava, ki je običajno nekoliko nadgrajen in prilagojen namenu in zasnovi danega sistema. Razlog za to enostavnost je, da mora PDM sistem nadzorovati različne komponente proizvoda, za razvoj in upravljanje le teh pa se lahko uporabljajo različne tehnologije in orodja. V primeru, ko PDM sistem vsebuje poseben modul, namenjen podpori upravljanja konfiguracij PO proizvodov (t.im. "SCM modul"), ta običajno vključuje višjenivojske oziroma naprednejše modele, ki so bolj prilagojeni posebnostim upravljanja procesov razvoja in vzdrževanja programske opreme.
SCM sistemi zelo poredko vsebujejo le model odjava/prijava, čeprav njihove funkcije navadno temelje na operacijah prijave in odjave. Uporabljajo bolj sofisticirane modele, ki se v zadnjem času, vsaj pri najkakovostnejših SCM sistemih, vse bolj usmerjajo v obvladovanje delovnih nalog in njim pripadajočih sprememb nad gradniki ali komponentami (moduli) izdelka. To hkrati pomeni, da študij transakcijskega modela in modela množic sprememb lahko predstavlja dobro izhodišče za razumevanje konceptov in mehanizmov, ki se vgrajujejo v najsodobnejše CM sisteme.

2.2.2 Taramova razvrstitev pristopov k upravljanju konfiguracij vloženih računalniških sistemov

Taramaa (Taramaa, 1998, str. 79) deli upravljanje konfiguracij za vložene računalniške sisteme v tri nivoje (glej sliko 15):

- *rešitve na področju razvoja verzij*, ki zagotavljajo ponovljiv nadzor nad verzijami. Vsebujejo tudi funkcije v zvezi z izdajanjem PO;
- rešitve *v izdajanje usmerjenega upravljanja konfiguracij PO* se uporabljajo v razvojnih sistemih z mnogoterimi in zapletenimi programskimi komponentami. Usmerjajo se na upravljanje različnih tipov konfiguracij in zagotavljanje pouporabljivosti komponent;
- pogoj za uporabo *v spremembe usmerjenega upravljanja konfiguracij PO* je ponovljiv, standardiziran proces izdelave PO. V takšen sistem se nato vgradijo funkcije za sledenje problemov, ki predstavljajo temelj za uvedbo učinkovitega upravljanja sprememb.

Slika 15: Razvrstitev pristopov k upravljanju konfiguracij vloženih sistemov



Vir: Taramaa, 1998, str. 80

2.2.2.1 Nadzor verzij

Nadzor verzij zagotavlja sredstva za vzdrževanje instanc sistemov, razvitih za različna okolja. Zagotovljeno je odlagališče podatkov in metapodatkov. Obstajajo formalizirani postopki za identificiranje konfiguracij in njenih gradnikov. V kolikor se v danem podjetju vodi več razvojnih projektov, je potrebno težiti za tem, da se ti formalizirani postopki vodijo na čim bolj enoten način. To je še posebej pomembno v primerih, ko se v različnih projektih uporabljajo isti deli PO. Ta pristop vodi do t.im. *standardiziranega in ponovljivega nadzora verzij*.

2.2.2.2 V izdajanje usmerjeno upravljanje konfiguracij programske opreme

Ta pristop se usmerja v izdelavo programske opreme. Rešitve obstajajo na različnih ravneh, od nestandardnih ročnih prijemov do celovitih avtomatiziranih rešitev:

- *ponovljivo izdelovanje programske opreme* se usmerja v izdajanje PO. Glavna usmeritev je v povezovanju nadzora verzij z upravljanjem komponent PO, posebej za gradnjo konfiguracij in izpeljanih (deriviranih) objektov. Najsplošnejši način obvladovanja konfiguracij je z oznakami (angl. labels). V Unix-ovem okolju so pogoste rešitve, ki temelje na orodju make. Z vidika vložene PO je največja prednost tega pristopa v nadzoru zapletenega procesa prevajanja in gradnje PO;
- *proizvajanje programske opreme skupaj z dokumentacijo.* Različni razvojni dokumenti, ki nastajajo med razvojem PO (npr. funkcijske specifikacije, rezultati načrtovanja, opisi vmesnikov med PO in SO itd.), so naravni sestavni del programske opreme. Različna dokumentacija nastaja sproti v procesu razvoja in vzdrževanja PO. Hkratno vključevanje programskih gradnikov in dokumentacije v izhodiščne konfiguracije zahteva razmeroma visoko raven discipline v procesu. Ta pristop se uporablja v naprednejših procesih razvoja in vzdrževanja PO (Taramaa, 1998, str. 84);
- *avtomatizirana izdelava PO za široko uporabo.* Ta pristop se posebej usmerja v zagotavljanje in optimizacijo pouporabljivosti komponent izdelka. Pouporabljivost lahko izboljšujemo na različnih nivojih, od prijemov pri načrtovanju zgradbe sistemov do prijemov pri kodiranju, selektivnega prevajanja ipd. Podobno kot zahteve po pouporabljivosti na upravljanje konfiguracij vpliva tudi uporaba sodobnejših življenjskih ciklov proizvodov in njihovih sestavnih delov, npr. prototipiranje - rezultat prototipiranja je lahko komponenta, ki jo je potrebno združiti z ostalimi komponentami sistema.

2.2.2.3 V spremembe usmerjeno upravljanje konfiguracij programske opreme

Ta pristop se usmerja v nadziranje oziroma upravljanje sprememb. Upravljanje sprememb je sicer pogosto konceptualno ločeno od upravljanja konfiguracij in vključuje proces razreševanja problemov. Vendar pa sta oba procesa, tako upravljanje konfiguracij kot upravljanje sprememb, v najnaprednejših razvojnih okoljih implementirana na visokem nivoju in neločljivo povezana. Tudi tu obstajajo različni nivoji rešitev, od delnih do celovitih:

- *upravljanje sledenja problemov.* Sledenje problemov je prva raven v spremembe usmerjenega SCM. Rešitve variirajo od enostavnih (npr. datotečnih) zapisov o opaženih nepravilnostih do uporabe obsežnih podatkovnih baz, kjer se vodijo povezave med problemi in aktivnostmi upravljanja konfiguracij. V enostavnejšem primeru se aktivnosti običajno izvajajo ročno, medtem ko so v primeru uporabe baz podatkov uporabljeni prijemi običajno del širše rešitve upravljanja konfiguracij. Sodobnejši CM sistemi običajno vsebujejo funkcije za sledenje problemov. S tem so običajno povezani tudi mehanizmi za izboljšanje procesa, saj se podatkovna baza za sledneje problemov običajno uporablja kot vir podatkov za uporabo procesnih metrik, ki se uporabljajo med upravljanjem razvojnega procesa;
- *z zahtevami vodeno upravljanje sprememb* zagotavlja mehanizem za povezavo med nadzorom sprememb in izdelavo programske opreme. Sistem za podporo upravljanju sprememb je lahko ali kupljen sistem ali v samem podjetju nadgrajena baza podatkov, ki beleži in vodi zapise o zahtevah za spremembo;
- *upravljanje aplikacij* najceloviteje obravnava proces vzdrževanja sistemov. Na tej ravni se uporabljajo posebna orodja za izvajanje posamezne aktivnosti vzdrževalnega procesa, npr. za: analiziranje oziroma razumevanje problema, njegovo lokalizacijo, ugotavljanje njegovega vpliva in analizo rešitve. Ta orodja se lahko v večji ali manjši meri uporabljajo že na nižjih, zgoraj opisanih nivojih, vendar so običajno tam nepovezana. Druga pomembna razlika v primerjavi s prejšnimi ravnmi je v tem, da upravljanje aplikacij vključuje rešitve na nivoju samega procesa. Ta pristop je procesno orientiran. Procesi se modelirajo in upravljajo. Pri tem je zaželjeno, da je proces kar se da *podatkovno usmerjen*, kar pomeni, da je kar se da transparenten in da uporabnika vodi v različnih možnih situacijah, ki se pojavljajo v procesu. V nasprotju s tem *v nadzor usmerjeni procesi* temeljijo na strogi formalnosti in omejitvah.

2.2.2.4 Upravljanje podatkov o proizvodih (Product Data Management)

Lahko ga obravnavamo v dveh ravneh, tako na ravni upravljanja komponent izdelka (npr. za PO), kot na ravni upravljanja celotnih proizvodov. Poseben izziv predstavlja povezovanje in prenašanje podatkov med različnimi PDM in SCM sistemi v geografsko porazdeljenem okolju. Pri tem so v praksi najpogostejše naslednje možnosti (Taramaa, 1998, str. 87): posamezen podporni sistem je nameščen na eni lokaciji; lahko je porazdeljen oziroma delujoč na večih lokacijah; obstaja tudi možnost sistema, sestavljenega iz SCM, PDM, MRP sistema in z njimi povezanih razvojnih okolij, ki kot povezan sistem delujejo v porazdeljenem razvojnem sistemu.
Poleg PO vloženi sistemi vsebujejo tudi druge komponente, kot npr. elektronske in mehanske dele. Pri njihovem nastajanju se uporabljajo različna računalniška okolja in metode. Pojavljajo se izzivi, npr. kako učinkovito povezati vse to v celovit sistem. Pri upravljanju konfiguracij za elektronske komponente izdelkov obstajajo posebnosti glede vsebine gradnikov (prevladujejo opisi simulacij) in glede postopka izdelave celotnih komponent, ki temelji na jezikih za opis SO (Taramaa, 1998, str. 87). Posebnosti v zvezi z obravnavo in ažuriranjem opisov SO in simulacij narekujejo posebne zahteve v zvezi z upravljanjem sprememb elektronskih komponent. Na tem področju se je kot zelo aktualna pokazala uporaba principov *sočasnega inženiringa*.

2.2.2.5 Globalno upravljanje proizvodov

Pri razvoju kompleksnejših izdelkov, katerih bistveni del predstavljajo vloženi računalniški sistemi, je povsem običajno, da se različne komponente izdelkov načrtujejo in razvijajo na različnih geografskih lokacijah. S tem je nastala potreba po zagotovitvi mehanizmov upravljanja konfiguracij, ki bi delovali v geografsko porazdeljenem razvojnem sistemu. T.im. *koncept virtualne organizacije*, vpeljan v 90. letih prejšnjega stoletja, predstavlja eno od rešitev pri globalnem upravljanju proizvodov (Taramaa, 1998, str. 90).

3. Analiza obstoječega stanja v podjetju

3.1 Funkcije obstoječega sistema in značilnosti procesa

3.1.1 Izdelki SI2000 in njihov življenjski cikel

Kot že omenjeno v uvodu, podjetje Iskratel razvija in trži telekomunikacijske izdelke SI2000. Neprekinjeno porajajoče se zahteve za te izdelke povzročajo diverzifikacijo in večanje števila proizvodov. Posledica je, da se v podjetju ne ukvarjamo s posameznimi izdelki, temveč z družinami podobnih proizvodov. Glavni viri funkcionalnosti izdelkov SI2000 so:

- *kupci oziroma trg* predstavljajo najpomembnejši vir zahtev po novih funkcionalnostih;
- *dobavitelji* običajno posredno vplivajo na nove funkcionalnosti, predvsem zaradi izboljšav ali zastaranja strojne ali programske opreme izdelkov;
- *nove tehnologije*, ki se pojavljajo v telekomunikacijah in z njimi povezanih panogah;
- spremljanje *konkurence* in njenih novosti na različnih trgih.

Razvoj omenjenih izdelkov, katerih glavni sestavni deli izdelkov so programska oprema, strojna oziroma materialna oprema (mehanski deli in elektronika) in uporabniška dokumentacija, se izvaja v večprojektnem okolju. Za industrijo telekomunikacij so značilni hitri cikli sprememb. Sposobnost sledenja hitrega tempa sprememb tako pomeni za podjetje enega ključnih dejavnikov poslovne uspešnosti. Do zdaj je strategija trženja izdelkov v veliki meri temeljila na izkoriščanju priložnosti, ki jih ponujajo tržne niše, kar pa od razvojnega okolja zahteva še dodatno prilagodljivost. Razvoj izdelkov se v glavnem izvaja na domači lokaciji, nekaj pa ga teče na drugih geografskih lokacijah.

Obstoječ način razvoja in vzdrževanja izdelkov SI2000, ko vsak izvajalec hkrati dela na večih proizvodih, zahteva od izvajalcev veliko učinkovitost pri delu. V vseh razvojnih področjih se večina razvoja PO izvaja z načrtovanjem in implementacijo sprememb na PO, razviti v prejšnjih projektih. Del sprememb se izvaja s paralelnim načinom dela, ko na istih gradnikih dela po več razvijalcev hkrati. Večina sprememb na izdelkih se običajno izvede ločeno po razvojnih področjih. Manjši del sprememb pa zadeva po več razvojnih področij hkrati. V takšnih primerih nastane potreba po koordinaciji projektov. Koordinacija tovrstnih aktivnosti se izvaja na vsaj dveh ravneh, tako na višjem nivoju, ki zadeva ključne vidike poslovanja podjetja in trženja izdelkov, kot tudi na nižjem nivoju, kjer se delegirajo posamezne naloge in določajo potrebni viri za dane aktivnosti.

Med razvojnimi platformami (operacijskimi sistemi) prevladujeta HP-UX (različica operacijskega sistema Unix, razvita s strani podjetja Hewlett-Packard), ki se uporablja na sistemih HP serij 700 in 800, in Windows, ki se uporablja na osebnih računalnikih. V zadnjih letih se je začelo v podjetju razmišljati tudi o operacijskem sistemu Linux, vendar se trenutno v razvojne namene še ne uporablja. Za razvoj in implementacijo posameznih sestavnih delov proizvodov se uporabljajo naslednja delovna okolja:

- za programsko opremo (operacijski sistem, aplikacije, upravljalna vozlišča...): VxWorks, pSOS, NewEra, Microsoft Visual Studio, Telelogic Geode SDL, C, C++, Java itd.;
- za razvoj mehanskih konstrukcij: SDRC IDEAS;
- za razvoj elektronike oziroma integriranih vezij: Mentor Graphics;
- za pisanje uporabniške dokumentacije: Microsoft Word, Frame Maker, Microsoft Visio, Microsoft Photo Editor, Canvas, Acrobat itd.

Za upravljanje življenjskega cikla proizvodov SI2000 se uporablja model vodnega slapu (angl. waterfall), ki je izpeljanka klasičnega življenjskega cikla. Življenjski cikel izdelkov je razdeljen na mejnike izdelka. Ta cikel lahko na zgoščen in poenostavljen način predstavimo takole:

1. *identificiranje tržne priložnosti, analiza in definiranje izdelka (do mejnika B200),* v kateri se: (a) zaznajo in poistovetijo tržne priložnosti; (b) določi in specificira proizvod, na podlagi podanih in usklajenih zahtev; (c) pripravi razvojni projekt, vključno s planom, ki vključuje časovni razpored izvajanja aktivnosti. Identificiranje novih zahtev za funkcionalnosti izdelka poteka neprekinjeno;
2. *razvoj izdelkov (med B200 in B600).* Med mejnikoma B200 in B600 se izvede razvojni projekt za dani proizvod. Projektni vodja spremlja in usklajuje delo razvijalcev in drugih udeležencev projekta ter ima glavno besedo pri pomembnih odločitvah v zvezi s funkcionalnostmi izdelka. Potem, ko se izvedeta podrobno načrtovanje in realizacija izdelka, nastopi faza integracije izdelka, kjer se vse razvite komponente povežejo v celovit sistem. Fazi integracije sledi faza verifikacije in validacije proizvodov. Mejnik B600 pomeni zaključek te faze, hkrati pa tudi zaključek razvojnega projekta. Tedaj gre izdelek v poskusno obratovanje. Prevzame ga servisno področje;
3. *podpora izdelku* se začne z mejnikom B600, torej po uspešno izvedenem prevzemu proizvoda s strani servisnega področja, in se konča z mejnikom B900, ko se izdelek umakne s trga. V teh fazah je za proizvod odgovoren produktni vodja. Pri tem sodeluje s servisno mrežo. Med glavnima mejnikoma B600 in B700 je proizvod v poskusnem obratovanju. Produktni vodja odloča o uvrstitvi izvedenk proizvoda na teren in o reševanju problemov v zvezi z izdelki. Ob mejniku B700 gre produkt v redno proizvajanje. Med mejnikoma B700 in B800 se neomejeno razširja po terenu. Mejnik B800 pomeni odločitev prodajnega področja o ukinitvi izdelka. Ob doseženem mejniku B800 se produkt začne postopoma ukinjati oziroma umikati iz uporabe. B900 pomeni njegovo dokončno ukinitev. Med mejnikoma B800 in B900 produktni vodja poskrbi za ustrezno zamenjavo z novim proizvodom.

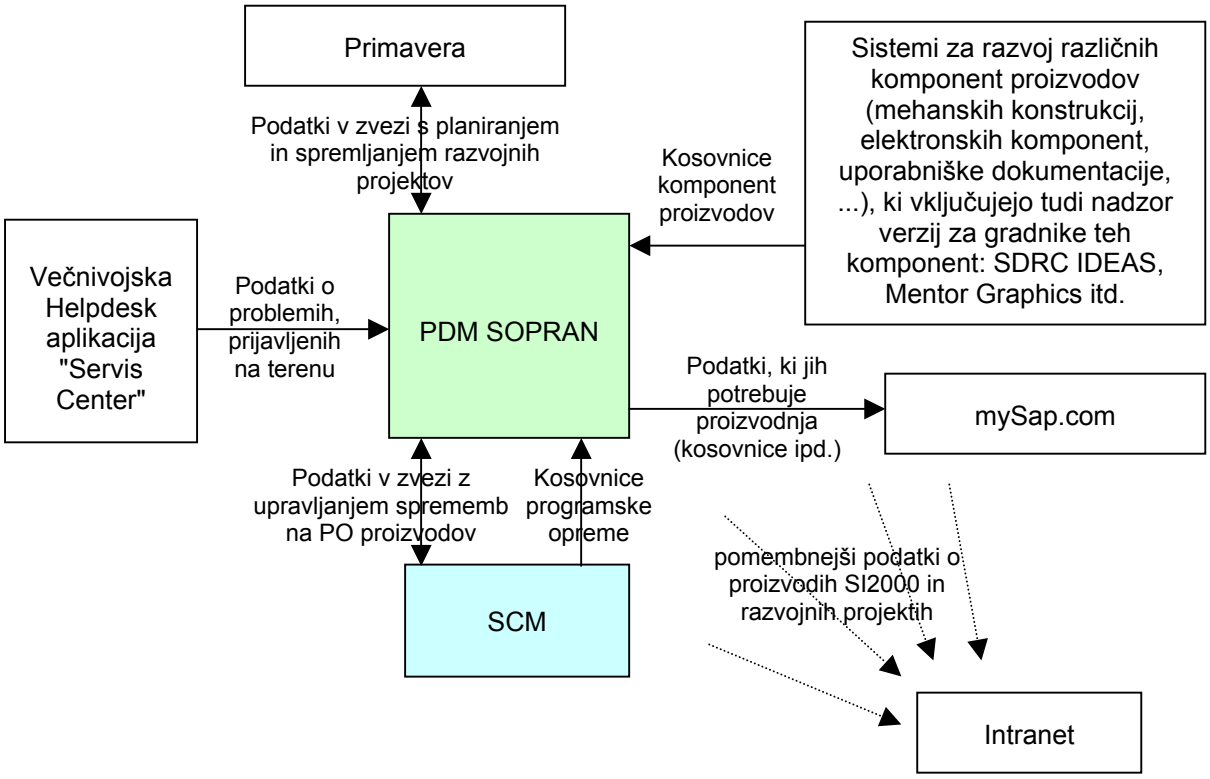
3.1.2 Sistemi za podporo upravljanju z življenjskim ciklom izdelkov in njihovimi konfiguracijami

3.1.2.1 Splošne lastnosti

Sistema, ki se trenutno uporabljata za upravljanje konfiguracij ter sprememb proizvodov in njihove PO, sta:

- PDM Sopran za upravljanje kosovnic proizvodov, upravljanje inženirskih sprememb in konfiguracij izdelkov, upravljanje sprememb komponent izdelkov (v primeru PO delno povezano z SCM sistemom), upravljanje kosovnic funkcij izdelkov, upravljanje dokumentacije;
- za upravljanje konfiguracij in sprememb programske opreme izdelkov SI2000 se uporablja doma razvit sistem SCM orodij.

Slika 16: Pomembnejše povezave in tokovi podatkov med sistemi za podporo življenjskega cikla produktov SI2000



Za vzdrževanje in administriranje vsakega izmed obeh sistemov je trenutno zadolžena posebna podporna skupina sistemskih inženirjev. Uporabnikov obeh sistemov je okrog 250. Za podporo upravljanju življenjskega cikla proizvodov SI2000 se uporabljajo tudi naslednji sistemi:

- geografsko porazdeljena aplikacija "Servis Center" (t.im. helpdesk) za prijavo pripetljajev na izdelkih, ki se uporabljajo na terenu;
- mySAP.com se v podjetju uvaja predvsem za področji financ in proizvodnje izdelkov. Poleg glavnega, standardnega modula R/3, so v kupljeni rešitvi zajeti tudi naslednji moduli: mySAP PLM za upravljanje z življenjskim ciklom proizvodov; mySAP SCM za upravljanje oskrbovalne verige za proizvode; mySAP CRM za upravljanje odnosov s kupci; skupna vhodna točka (portal). Pri tem se nekatere funkcije modulov podvajajo;
- na intranetu se objavljajo tekoči podatki o produktih, poteku projektov ipd.;
- za planiranje in spremljanje poteka razvojnih projektov se uporablja aplikacija primavera.

Poleg tega se v podjetju za označevanje in vodenje problemov na internih informacijskih sistemih, vključno z njihovo infrastrukturo, uporablja sistem Remedy.

V sistemih za podporo upravljanju konfiguracij proizvodov in njihove PO uporabljamo sestavljeno upravljanje konfiguracij (Frey-Pučko, 2002, str. 11), ki vključuje vse modele upravljanja konfiguracij, predstavljene v razdelku 2.2.1. Kombinacija modelov je nastala postopoma in z manjšimi nadgrajevanji. Postopki se izvajajo delno ročno in delno v orodjih.

Tabela 2: Modeli upravljanja konfiguracij proizvodov SI2000 in njihovih komponent

Tip upravljanih elementov	***Model***	***Proces***	***Orodje***
Celotni proizvodi	odjava/prijava	Gradnja, povezovanje in izdajanje kosovnic produktov	SOPRAN
Programska oprema	odjava/prijava	Izdajanje, potrjevanje in arhiviranje gradnikov v testni in uradni knjižnici, gradnja in izdajanje kosovnic PO	Lastni SCM, SOPRAN
Materialna oprema	odjava/prijava	Izdajanje kosovnic materialne opreme	SOPRAN
Problemi in spremembe	transakcijski model, model sprememb, odjava/prijava, kompozicijski model	Vnos množice sprememb v kopijo konfiguracije programske opreme (model sprememb, transakcijski model), spremembe nad strukturo paketa (kompozicijski model), upravljanje zahtev za spremembo (odjava/prijava)	Lastno SCM orodje, SOPRAN
Dokumenti	odjava/prijava	Izdajanje, potrjevanje in arhiviranje dokumentov	SOPRAN, Intranet

Vir: Frey-Pučko, 2002, str. 11

3.1.2.2 PDM sistem SOPRAN

Ogrodje PDM sistema SOPRAN sestavlja sistem Sherpa PIMS, ki pa je bil pozneje prilagojen našim zahtevam. Sistem SOPRAN je namenjen upravljanju inženirskih sprememb nad izdelki SI2000, vodenju kosovnic izdelkov, kosovnic funkcij, kosovnic uporabniške dokumentacije, kosovnic materialne opreme in vrhnjih kosovnic programske opreme.
Sherpa PIMS se na tržišču ne prodaja več, podjetje Sherpa je bilo prevzeto s strani SDRC, ponudnika orodja Metaphase. Tega je lani prevzelo podjetje EDS, ki funkcionalnosti Metaphase-a vključuje v module svojega EDM/ERP sistema Teamcenter. Vzdrževanje Sherpe je torej odvisno le od interne skupine vzdrževalcev. Skozi ta sistem naj bi se izvajale vse zahteve naročnikov za izdelke in zahteve za spremembo izdelka v primeru problemov na terenu. SOPRAN oziroma sistem Sherpa PIMS je nekoliko podrobneje predstavljen v razdelku 5.2.2. Ena najmočnejših funkcij Sherpinega sistema je podpora upravljanja inženirskih sprememb na izdelkih (glej razdelek 3.1.3).

3.1.2.3 SCM sistem

SCM sistem se uporablja za upravljanje življenjskega cikla PO izdelkov SI2000, vključno z vodenjem konfiguracije PO izdelkov in podporo izdajanja, obravnave in arhiviranja programskih gradnikov ter podporo gradnje in izdajanja programskih paketov. Sistem je zgrajen okrog odlagališča podatkov, ki temelji na uporabi orodja SCCS in podatkovne baze Informix. SCM sistem je nameščen na večprocesorskem strežniku z operacijskim sistemom HP-UX. Sistem je neposredno dosegljiv z drugih Unixovih sistemov s pomočjo Unixovega mehanizma za povezovanje datotečnih sistemov (t.im. *mount*). Projekti, ki se vodijo na sistemih Windows, uporabljajo SCM sistem preko Sambe in X-emulatorjev (X je grafično okolje za Unix). SCM sistem ima grafični vmesnik, razvit z aplikacijo Uim/X. Ta ima omejene možnosti obvladovanja grafičnih objektov.

Osnovna funkcionalnost orodja SCCS, ki je en temeljev za naš SCM sistem, je predstavljena v razdelku 5.2.1. Ker za podporo tako kompleksnega procesa upravljanja konfiguracij programske opreme, kot se izvaja v Iskratelu, ta funkcionalnost ni zadostovala, je bila v preteklih letih razširjena na naslednja področja:

- uvedba posebne vrste tekstovnih datotek (t.im. *makrokrmilnih datotek*), v katerih so opisane konfiguracije programskih paketov. S Pomočjo te vrste datotek se je vzpostavila sledljivost nad konfiguracijami programskih paketov;
- uvedba posebnega tipa odlagališča (t.im. *testna knjižnica*) namenjenega podpori izdajanja testnih paketov. Značilnosti tega okolja sta: testno okolje se uporablja vzporedno uradnemu okolju; ko je dani gradnik PO zrel za uradno izdajo, se iz testnega okolja prenese v uradno okolje, kjer se tudi arhivira. Po uspešnem prenosu gradnika se njegovi podatki v testnem okolju zbrišejo; če prenos v uradno okolje ni uspešen (nadzorna komisija zavrne gradnik), se gradnik vrne v testno okolje do trenutka, ko so odpravljene pomanjkljivosti in ko je spet goden za prenos v uradno okolje;
- uvedba delovnih področij. Vsakemu projektu ali zaključeni sorodni skupini produktov, ki imajo enako razvojno metodologijo, je bilo prirejeno svoje delovno področje. Za vsako od njih obstaja lastno okolje upravljanja konfiguracij PO;
- izvedena je bila nadgradnja SCCS sistema z BP Informix on-line s podporo za delo v mreži;
- izvedena je bila improvizirana podpora za delo z binarnimi datotekami;
- izvedena je bila podpora procesa sprememb gradnikov PO. V ta namen je bilo izdelanih nekaj orodij, namenjenih podajanju zahteve za obravnavo spremenjenega gradnika, obravnavi gradnika, prenosu iz testne v uradno knjižnico ipd.;
- dodana so bila orodja, namenjena administriranju podatkov in metapodatkov v SCM sistemu, in orodja, namenjena pregledovanju vsebine gradnikov v odlagališču;
- varnostni mehanizmi: zagotovljena je bila ustrezna zaščita datotek pred nepooblaščenim doseganjem, prav tako pa tudi redna, periodična izdelava kopij programske opreme;
- kodni sistem za gradnike PO je podedovan z IBM-ovega centralnega računalnika. Za gradnike se uporabljajo 8-mestne kode, od česar so štiri mesta v kodi rezervirana za osnovno identifikacijo gradnika, dve mesti za opis vejitve gradnika, dve mesti pa za opis verzije gradnika.

Poleg pravkar predstavljenega *uradnega SCM sistema* obstaja tudi t.im. *'poluradni' SCM sistem*, ki je sestavljen iz skriptov (programov), napisanih v jeziku operacijskega sistema HP-UX. Ti skripti se navadno uporabljajo za pridobivanje informacij iz uradnega SCM sistema. Poluradni SCM sistem vzdržujejo koordinatorji programskih paketov, pri čemer jim pomagajo sistemski inženirji, zadolženi za uradni SCM sistem. Poleg omenjene skupine ga uporabljajo tudi mnogi razvijalci pri svojem delu.

3.1.2.4 Povezave med PDM in SCM sistemom

V preteklem letu sta se z namenom izboljšati nadzor spreminjanja programske opreme izdelkov med Sopranom in SCM sistemom začela vzpostavljati dva tipa povezav:

- *povezava "proizvod-programski gradnik".* S to povezavo se vsaka nova verzija katerega koli programskega gradnika ob predajanju v obravnavo poveže s proizvodi, za katere je bila izdelana. Povezava torej omogoča, da za vsako novo izdajo proizvoda dobimo nabor spremenjenih verzij gradnikov, ki jih bomo uvrstili v to izdajo, in obratno, za vsako verzijo programskega gradnika imamo na voljo informacije, v katere proizvode jo bomo predvidoma uvrstili. Povezava je realizirana s podvajanjem podatkov med sistemoma oziroma prenosom podatkov o proizvodih iz PDM sistema v SCM sistem;

- *povezava "ECR-programski gradnik".* Ta naj bi povezala spremembe v Sopranu s posameznimi gradniki v SCM sistemu. S to povezavo naj bi za vsak ECR (t.j. zahteva za spremembo) dobili seznam gradnikov, ki ta ECR rešuje, in obratno: za vsak programski gradnik (paket) bi s tem dobili seznam ECR-ov, ki so bili rešeni v tem gradniku (ali paketu). Tudi ta vrsta povezave je realizirana s podvajanjem podatkov oziroma s prenosom podatkov o ECR-ih iz Soprana v SCM sistem.

3.1.3 Ključne značilnosti procesa upravljanja konfiguracij in sprememb izdelkov in njihove PO

3.1.3.1 Ključne vloge v procesu

Ključne vloge, tako v procesih razvoja in vzdrževanja izdelkov kot v procesu upravljanja njihovih konfiguracij in sprememb, so:

- *produktni vodja* upravlja tako tržni kot tehnični vidik izdelkov. Ovrednoti zahteve za spremembo in načrtuje spremembe na izdelkih, v skladu z zahtevami kupcev in ostalih predlagateljev sprememb;
- *projektni vodja* poskrbi za planiranje, koordiniranje, spremljanje in zagotavljanje optimalnih pogojev za izvedbo razvojnih projektov;
- *projektni koordinator* je v času poteka projekta pomočnik vodju projekta. Za vsak projekt in vsako razvojno področje se delegira posebej. Projektni koordinator je odgovoren za nadzor programskega paketa posameznega projekta v vseh razvojnih fazah projekta; nadzira programski paket v razvojnih fazah; nadzira testne in uradne izdaje programskega paketa; odgovoren je za kakovost programskega paketa itd.;
- *vodja posameznega razvojnega področja* skrbi za tehnološki vidik in zgradbo tiste komponente sistema SI2000, katero razvija njegovo razvojno področje. V skladu s tem skrbi za metodologijo, ki se uporablja tako v razvojnem in vzdrževalnem procesu kot v procesu upravljanja konfiguracij izdelkov in njihove programske ter strojne opreme;
- *vodja razvojne službe* podrobno planira aktivnosti po izvajalcih. Potek planiranih aktivnosti usklajuje z ostalimi vodji služb v razvojnem področju in z vodjem razvojnega področja. Poroča o realizaciji aktivnosti, ki jih izvajajo izvajalci v njegovi službi. Skrbi za izvajanje predpisane metodologije in tehnologije razvojnega dela v službi;
- *razvijalec oziroma vzdrževalec* poskrbi za razvoj ali realizacijo sprememb na izdelkih, v skladu s planiranimi aktivnostmi;
- *koordinator programskih paketov* skrbi za sestavljanje, gradnjo in izdajanje programskega paketa za dani izdelek. V ta namen koordinira dejavnosti vseh ostalih udeležencev projekta;
- *administrator projekta* izvaja operativna dela v procesu upravljanja sprememb in konfiguracij izdelkov: sestavlja vrhnje kosovnice izdelkov, izdeluje izhodiščne konfiguracije izdelkov, poverja naloge izvajalcem, v sodelovanju s produktnimi vodji izdeluje sezname problemov, katerih rešitve naj bodo uvrščene v naslednjo izdajo proizvodov ipd.;
- *administrator sistema* opravlja administrativne posege na sistemu za upravljanje konfiguracij;
- *sistemski inženir* poskrbi za razvoj, vzdrževanje in izboljšave sistemov za upravljanje konfiguracij izdelkov in vseh drugih sistemov za podporo poslovanja, vključno z ustrezno informacijsko infrastrukturo.

3.1.3.2 Proces upravljanja inženirskih sprememb na izdelkih

V nadaljevanju je v kratkem opisan proces upravljanja inženirskih sprememb na izdelkih, ki predstavlja enega najpomembnejših procesov pri upravljanju izdelkov SI2000. Proces je v veliki

meri avtomatiziran s PDM sistemom SOPRAN. Začne se s prijavo zahteve za spremembo (ECR-a) v sistem. Ta zahteva je lahko podana na podlagi opažene pomanjkljivosti na danem izdelku (med rednim obratovanjem na terenu ali med fazo preverjanja in vrednotenja) ali pa gre za zahtevo po novi funkcionalnosti izdelkov. Zahteve za spremembo se s povezavo tipa *"raised against"* (glej sliko 17) povežejo na tisto verzijo kosovnice proizvoda, na katero se zahteva nanaša (t.j. v kateri se je pojavila napaka ali kateri naj se doda nova funkcionalnost). Te kosovnice so razgrajene (povezava tipa *"uses"*) na podkosovnice. S tem je določena celotna struktura proizvoda.

Po evidentiranju prijavljenih zahtev za spremembo se izvedejo administrativne, poslovne in tehnične analize. Administrativne analize vključujejo preglede prijav in ugotavljanje, ali so se te prijave že kdaj obdelale (ali gre za podvojene prijave) ipd. Poslovne analize morajo odgovoriti na vprašanja kot npr.:

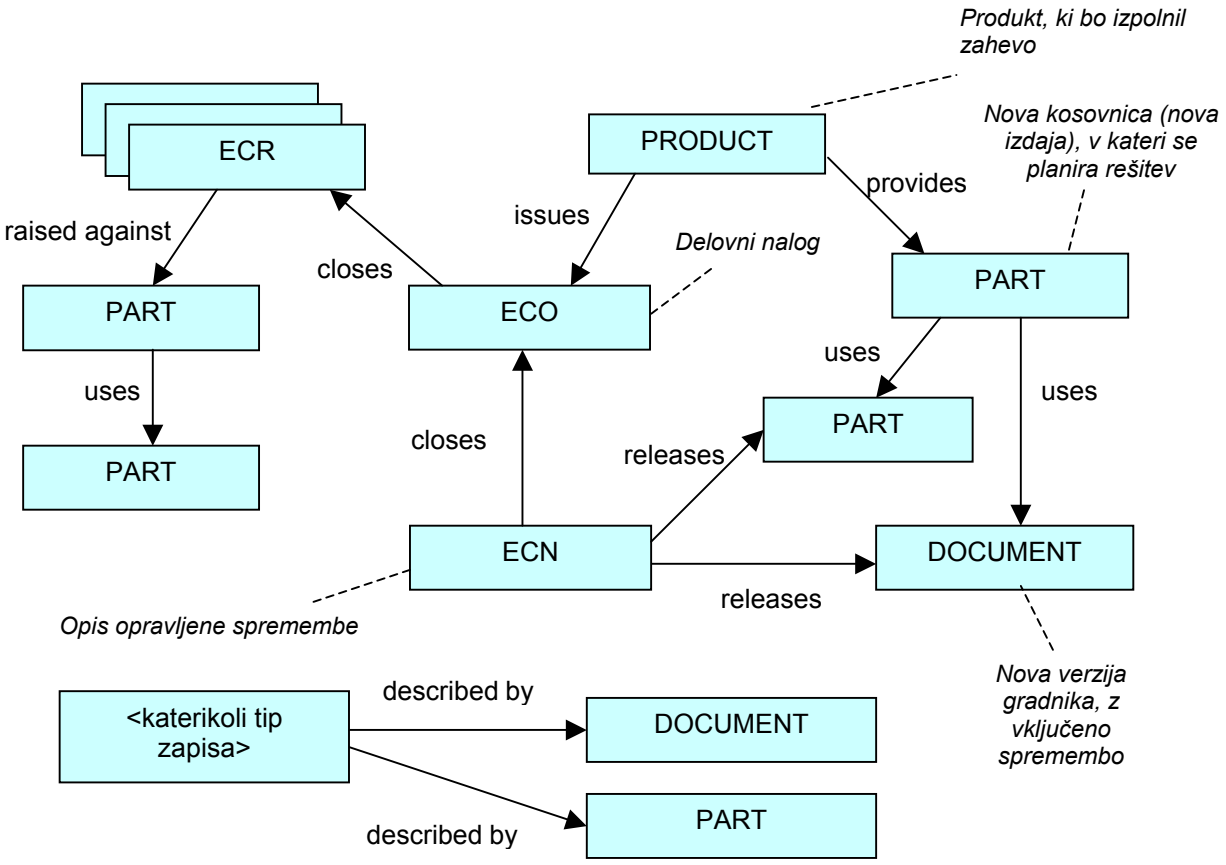
- Za kako pomembno spremembo na izdelku gre? Ali je sprememba potrebna? Ali je potrebno spremembo izvesti takoj?
- Kaj nam bo ta sprememba prinesla? Koliko virov bomo za to porabili?

Tehnična analiza je namenjena čim boljši podpori in čim kakovostnejši izvedbi poslovne analize in zadeva predvsem vprašanja tipa:

- V katere proizvode so vključeni gradniki in kosovnice, povezani z dano spremembo?
- Kako bo sprememba vplivala na zgradbo izdelkov?
- Kakšne spremembe nad gradniki bo potrebno izvesti?

Po izvedenih analizah se sprejme poslovna odločitev glede izvedbe spremembe in njene pomembnosti (prioriteti), zagotavljanja virov za izvedbo spremembe, določanja rokov za izvedbo ipd. Kreira se delovni nalog (ECO) za izvedbo spremembe in se poveri izvajalcu. Sprememba bo vključena v nove verzije kosovnic proizvoda, ki so z zapisom tipa PRODUCT povezane s povezavo tipa *"provides"*. Sprememba bo zagotovljena z ustrezno izvedbo delovnega naloga za dani proizvod - povezava *"issues"*. Po zaključeni delovni nalogi izvajalec opiše izvedeno spremembo oziroma napiše delovno poročilo (ECN), ki se s povezavami tipa *"releases"* nanaša na spremenjene gradnike in kosovnice. Izdajanju in arhiviranju gradnikov sledijo (v skladu s projektnimi in/ali drugimi plani, v katerih so načrtovane izdaje novih verzij proizvodov) izdajanja kosovnic komponent in celotnega izdelka. Po potrjeni rešitvi oziroma potrjenih spremembah se bo vzpostavila povezava tipa *"closes"*, ki kaže od delovnega poročila proti delovnemu nalogu in od delovnega naloga proti zahtevi za spremembo. Med samim izvajanjem spremembe izvajalec ali kdor koli drug lahko izdela dodatne dokumente ali kreira dodatne kosovnice gradnikov, ki jih lahko preko povezave *"described by"* poveže na tiste verzije entitet, na katere se nanašajo.

Slika 17: Planiranje in izvedba sprememb



Vir: Interna dokumentacija podjetja

3.1.3.3 Značilnosti metodologije in procesa upravljanja konfiguracij PO

Nekaj ključnih značilnosti metodologije in procesa upravljanja konfiguracij in sprememb programske opreme izdelkov SI2000:

- trenutna razvojna metodologija, ki neposredno vpliva na proces upravljanja konfiguracij PO, je utemeljena na spreminjanju in izdajanju tako posameznih gradnikov kot komponent (npr. programskih paketov) izdelkov;
- ob začetku kakršnega koli novega razvojnega ali vzdrževalnega projekta razvijalci kreirajo svoja delovna okolja. To store s kopiranjem datotečne strukture izdanih programskih paketov v svoj datotečni sistem. Omenjeno datotečno strukturo po vsaki izdaji na podlagi izhodiščnih konfiguracij pripravijo koordinatorji programskih paketov na ustreznih, temu namenjenih sistemih;
- programske gradnike oziroma komponente, ki jih izdajo izvajalci sprememb, pred shranitvijo v odlagališča podatkov obravnavajo (pregledajo in potrdijo ali zavrnejo) odbori za nadzor sprememb na PO. Sestavljeni so iz vodij organizacijskih enot, zadolženih za razvoj PO. Naloga odbora je formalna obravnava in sprejem odločitev glede ustreznosti programskih gradnikov. Gradnjo in izdajanje PO izvedejo koordinatorji programskih paketov. Ko so programski paketi izdani, se njihove izhodiščne konfiguracije uvrste v vrhnje kosovnice izdelkov SI2000, kar store projektni administratorji. Temu sledi še sestavljanje, obravnava in potrjevanje (lahko tudi zavračanje) kosovnic proizvodov, pri čemer se v te kosovnice poleg PO vključijo še kosovnice, ki določajo druge komponente izdelkov (mehanski in elektronski deli, uporabniška dokumentacija ipd.). S tem so izdelki

pripravljeni za obdelavo v naslednjih fazah procesa (izvajanje sistemskih testiranj, proizvodnja, testno obratovanje sistemov na terenu ipd.);
- v fazah podpore proizvodom se rešujejo problemi na proizvodih. Analizirajo, načrtujejo in realizirajo se tiste zahteve za spremembo proizvodov, ki so planirane za ustrezno vzdrževalno izdajo izdelka.

3.2 Kritična analiza obstoječega stanja

Med poznavalci trenutne situacije v podjetju prevladuje visoka stopnja strinjanja glede tega, da je en glavnih vzrokov za obstoječe in v nadaljevanju navedene pomanjkljivosti procesa upravljanja konfiguracij in sprememb proizvodov predvsem v tem, da tega procesa, vključno z metodologijami in podpornimi sistemi, ki se v njem uporabljajo, nikoli nismo celovito upravljali oziroma ga obravnavali kot nekaj, kar bi moralo delovati kot celota. Trenutno stanje je rezultat dograjevanj in delnih rešitev skozi celotno zgodovino upravljanja konfiguracij v razvoju in vzdrževanju proizvodov SI2000. Za razmeroma veliko število problemov, evidentiranih v raziskavah procesnega področja "upravljanje konfiguracij", ki so bile v podjetju izvedene v preteklih treh letih, je značilna tudi ugotovitev, da so jih uporabniki prepoznali kot probleme upravljanja konfiguracij, dejansko pa imajo izvor v drugih procesih, ki so z upravljanjem konfiguracij neposredno povezani, npr.: način upravljanja produktne linije in hkraten razvoj mnogih različic proizvodov; delo enega razvijalca na mnogih vzporednih projektih; problemi na področju upravljanja zahtev in neprestano, premalo nadzorovano vletavanje novih zahtev za proizvode v proces; še več pozornosti bi morali posvetiti obvladovanju zgradbe izdelkov, t.j. zmanjšanju zapletenosti povezav in odvisnosti med različnimi komponentami izdelka, ipd.
Kljub omenjeni ugotovitvi pa je tudi v samem procesu upravljanja konfiguracij in sprememb proizvodov, vključno z metodologijo in podpornimi sistemi, ki se v njem uporabljajo, razmeroma veliko priložnosti za izboljšavo. Mnogi od ugotovljenih problemov se dejansko že rešujejo, k omenjenim problemom pa se pristopa vse bolj sistematično in celovito.

3.2.1 Razvojni proces in upravljanje konfiguracij izdelkov

V skladu z mnogokrat preverjeno ugotovitvijo, da je en glavnih pogojev za izvajanje izboljšav poslovnega procesa izčrpna in razumljiva predstavitev le-tega (Neuendorf, 1998, str. 20), je bilo v preteklih letih izdelanih več analiz procesa upravljanja konfiguracij izdelkov in njihovih komponent, vključno s podpornimi sistemi. Namenjene so bile predvsem izvajanju izboljšav na tem procesnem področju.
Celoten koncept organiziranosti, ki je različica matrične organiziranosti, je privedel do tega, da so razvojna področja postala precej avtonomna pri določanju metodologije svojega dela. Ker ta "proces osamosvajanja" ni bil v zadostni meri koordiniran in nadzorovan, je s tem prišlo do razbitosti procesa razvoja PO. V enakih okoliščinah v različnih področjih prihajajo do različnih rezultatov. Ni enotnega razvojnega procesa, kar dodatno otežuje učinkovit pregled oziroma nadzor nad potekom razvoja. V zadnjem času se stanje izboljšuje, med drugim s pomočjo uporabe modela CMM (ocenjevanje zrelosti razvojnega procesa) in uvajanjem celovitih procesnih pristopov.
Razvojni in vzdrževalni proces za izdelke SI2000 je razmeroma dobro določen, vendar pa ni dovolj dosledno voden oziroma upravljan - lahko pa rečemo, da je tako ali drugače "koordiniran", temu so prilagojene tudi vloge v procesu. Tako se marsikatera aktivnost "v posebnih primerih" brez večjih težav obide. Hkrati se vzdržuje razmeroma veliko število

proizvodov in njihovih različic (nekaj desetin, njihovo število pa še narašča), pri čemer se uporablja paralelni razvoj. Velika stopnja odprtosti v razvoju SI2000 po eni strani omogoča izjemno prilagodljivost, relativno ugoden "time-to-market" in veliko prostora za interne postopke, po drugi strani pa postavlja visoke zahteve za učinkovito obvladovanje procesov. Z množenjem izdelkov postaja proces upravljanja sprememb in konfiguracij izdelkov vse težje obvladljiv (z obstoječimi metodami in podpornimi sistemi), kar lahko predstavlja vse večjo oviro pri upravljanju njihove kakovosti. V zvezi z organizacijo in porazdelitvijo vlog ter odgovornosti veljajo poleg zgoraj navedenega naslednje ugotovitve:

- razvojni proces je določen na makro nivoju, na izvedbenem nivoju so definirani samo nekateri postopki oziroma so definirani za razvojno okolje dela produkta (Frey-Pučko, 2002, str. 9). Proces je izjemno kompleksen, ima veliko izvedbenih variant v različnih okoljih, ki so pomanjkljivo definirane. Nihče od udeležencev ne pozna celotnega procesa v podrobnosti. Odgovornosti v procesu niso jasno določene, velika teža pri odločanju je na razvijalcih;
- pomanjkanje planiranja upravljanja konfiguracij in pomanjkljivo določene odgovornosti v zvezi s tem procesnim področjem je pripeljalo do tega, da imamo sive cone v procesu, ki jih slabo obvladujemo. Ni jasno, kdo je za nekatera področja (npr. za različne kodne sisteme) v celoti pristojen, zato se reševanje nekaterih ključnih problemov v procesu in sistemu vleče dalj časa, kot bi bilo optimalno;
- ločenost služb za sistemsko podporo, razvoj in vzdrževanje SCM in PDM sistemov je v preteklosti prinesla precej negativnih učinkov, predvsem: podvajanje funkcionalnosti v obeh sistemih, pomanjkljiva povezanost obeh sistemov in slabša koordinacija na področju upravljanja življenjskega cikla produktov SI2000. Zato se zdaj daje vse večja pozornost povezanemu, usklajenemu delovanju obeh skupin.

Za proces upravljanja sprememb in konfiguracij izdelkov SI2000 in njihove programske opreme je značilno, da je vpet v vse faze življenjskega cikla teh izdelkov. Omenjeni proces v splošnem teče solidno, ima pa nekaj šibkih točk:

- trenutne aktivnosti upravljanja konfiguracij se v razmeroma veliki meri izvajajo z ustno ali neformalno koordinacijo in se vrte okrog nekaterih posameznikov, kar lahko povzroča zastoje in možnosti za kritične napake;
- proces upravljanja konfiguracij PO se izvaja v razvojnih okoljih za različne tipe PO z različnimi razvojnimi metodologijami in različnimi logičnimi pristopi k združevanju gradnikov PO (moduli). Enako velja za različnost pristopov k vključevanju gradnikov v programske pakete (uporaba vejitev ali skupnih gradnikov s pogojnim prevajanjem);
- obstajajo težave, povezane s spoštovanjem sprejetih postopkov. Primer: zaradi časovnih pritiskov se pregledi dokumentacije in izvorne programske kode ter postopki podrobnejših testiranj za določene primere obidejo ali premaknejo v naslednjo fazo življenjskega cikla danega izdelka. Več je potrebno narediti v zvezi z vključevanjem vodstva v reševanje tovrstnih problemov, izboljšati planiranje, nadzor in druga orodja upravljanja, ne nazadnje pa tudi zvišati raven avtomatizacije postopkov, da bi potegnili večji učinek iz dobrih praks, ki se že vodijo v podjetju;
- kot enega izmed pomembnejših problemov so uporabniki ocenili tudi neobstoječo podporo za izdajanje t.im. patch paketov. V zadnjem letu je bilo v zvezi s tem problemom sprejetih že nekaj ukrepov (uradno izdajanje patch paketov).

3.2.2 Trenutni podporni sistem

3.2.2.1 Splošne ugotovitve

V celotnem sistemu upravljanja konfiguracij je identificiranih večina elementov konfiguracije (Frey-Pučko, 2002, str. 9), niso pa dovolj identificirane povezave med njimi. V množici različic produktov npr. ni jasno vzpostavljene in pregledne povezave med zahtevami, produkti, paketi, gradniki in spremembami.

Velika težnja nekaterih predstavnikov vodstva je poenotiti nadzor nad spreminjanjem programske in materialne opreme proizvodov. To je v tem trenutku oziroma v dani situaciji razumljiva in logična zahteva, ni pa povsem očitno, katere zares bistvene pozitivne dolgoročne učinke bi s tem pristopom pridobili. Trenutno na tržišču ni na razpolago zares dobrih podpornih sistemov, ki bi to omogočali na ustrezni kakovostni ravni. Ocenjujem, da bi tovrstne probleme lahko precej učinkovito reševali tudi z bolj premišljenim upravljanjem s komponantami izdelkov.

Obstoječi podporni sistemi trenutno sicer ustrezajo obstoječim načinom dela, za kakovostni preskok na višji nivo učinkovitosti in kakovosti dela pa bi bilo vanje potrebno vložiti zelo veliko dodatnega dela. Iz omenjenega razloga so komercialni sistemi toliko bolj privlačni. Zahteva za obvladovanjem komponentno usmerjenega razvoja PO zožuje izbiro komercialnih sistemov na nekaj vrhunskih, med katerima sta npr. Telelogicov CM Synergy in Rationalov ClearCase.

Pričakuje se, da se bo jedro Iskratelovega poslovnega informacijskega sistema sčasoma preselilo v SAP. SAP bo zagotavljal krovni pogled nad poslovnim sistemom. V podjetju je že v tem trenutku opazna tendenca po prenosu funkcij za spremljanje in vodenje projektov v ta sistem (SAP trenutno v zvezi s podporo projektov pokriva stroškovni in logistični del projektov). Vendar ta sistem v podjetju verjetno ne bo nikoli v celoti podprl vseh področij poslovanja (težko bi v okviru SAP-a npr. našli dovolj dobro rešitev za razvoj PO), zato je integracija pomembna. Povezovalno infrastrukturo, ki bo osnova za vse integracije med sistemi za podporo poslovanju, bo v podjetju v prihodnje zagotavljal Microsoftov BizTalk. BizTalk poleg sinhronih podpira tudi asinhrone povezave, na katerih temelje poslovne povezave, omogoča pa tudi interaktivno delo.

3.2.2.2 PDM Sopran

PDM sistem Sopran ima nekaj dobrih lastnosti, ki omogočajo učinkovito upravljanje proizvodov SI2000 - predvsem v zvezi s strukturiranjem različnih podatkov in informacij o proizvodih ter na podlagi tega možnosti učinkovitega vodenja sprememb nad le-temi. Sistem ima tudi nekaj slabosti, ki onemogočajo izkoriščanje njegovega celotnega potenciala. Uporabniki vidijo največ pomanjkljivosti PDM sistema v: zapletenosti postopkov pri delu z orodjem; premajhni preglednosti nad podatki in premalo prijaznem uporabniškem vmesniku. Poleg tega sistem nima več zagotovljenega vzdrževanja s strani proizvajalca. Sherpa je bila prevzeta s strani podjetja SDRC, SDRC pa pozneje s strani EDS.

3.2.2.3 SCM sistem

V zvezi s trenutnim SCM sistemom se poleg dobrih lastnosti (predvsem njihovi prilagojenosti trenutnemu načinu dela in željam uporabnikov) pojavljajo naslednje pomanjkljivosti:

- sistem v zelo omejeni meri pokriva oddaljene razvojne lokacije;
- obstoječi SCM zagotavlja učinkovito podporo le za Unix;
- problem obravnavanja naraščajočega deleža binarnih datotek ostaja odprt;
- hitrost - odzivni časi orodij niso najboljši, kar je povezano tudi s tehnologijo SCCS-a;
- funkcionalnost upravljanja delovnih prostorov je podprta delno. Oorodja npr. ne omogočajo transparentnega načina dela na programskem paketu. Vsak uporabnik mora

ustvariti in ažurirati lastno kopijo paketa na svojem delovnem sistemu s kopiranjem, vsako ažuriranje pa zahteva precej časa;
- pomanjkljiv nadzor nad verzijami datotečnih map oziroma konfiguracijo paketa. Funkcionalnost je realizirana delno, preko t.im. makro steerov;
- pri gradnji programske opreme produktov ni mogoče pouporabiti že zgrajenih objektov;
- druge omejitve sistema SCCS, npr.: omejitve glede števila vrstic in dolžine vrstic; omejitve glede števila verzij v dani vejitvi (največ 99);
- omejitve obstoječega kodnega sistema; itd.

3.2.2.4 Povezave med sistemoma in perspektiva sistema

Pri obvladovanju sprememb na družini produktov je z vidika orodij največja ovira nepovezanost SCM in PDM sistemov (Frey-Pučko, 2002, str. 12). To so izpostavili tudi uporabniki v izvedenih anketah in intervjujih. Nekatere operacije in podatki med sistemoma se podvajajo, kar povečuje možnosti za nekonsistentnosti v sistemu. Vzdrževanje vzporednih funkcionalnosti je dražje, uporabniški vmesniki sistemov niso enotni, celotni sistem ni tako učinkovit kot bi bil lahko. Z ravnijo povezanosti sistemov je povezan tudi problem glede zagotavljanja sledljivosti v procesu upravljanja sprememb proizvodov. Ta se trenutno rešuje z uvajanjem povezav "ECR - programski gradnik" in "programski gradnik - proizvod". Implementacija obeh tipov povezav se kaže kot razmeroma kompleksna. Tudi uporaba SCM sistema je z uvedbo teh povezav postala bolj zahtevna. Te povezave so se do zdaj v sistem vnašale in vodile v glavnem ročno. Kljub opisanim omejitvam lahko tako PDM kot SCM sistem v osnovnih funkcionalnostih upravljanja konfiguracij zagotavljata podporo še nekaj let ob enako kakovostnem vzdrževanju in manjših dopolnitvah (Frey-Pučko, 2002, str. 12). Pomembno je, da se omejitev zavedamo in poskušamo njihove negativne učinke, dokler nimamo dolgoročne rešitve, sproti kompenzirati v procesu.

3.2.3 Stanje ostalih procesov v neposredni povezavi z upravljanjem konfiguracij izdelkov

Upravljanje konfiguracij kot rečeno ni tesno povezano le s procesom razvoja in vzdrževanja proizvodov, temveč tudi z nekaterimi drugimi procesnimi področji. En osnovnih namenov upravljanja konfiguracij je upravljanje rezultatov oziroma delnih proizvodov, ki nastanejo v procesih ostalih procesnih področjih, in sprememb teh rezultatov. Zato morebitne pomanjkljivosti ali neusklajenosti na ostalih procesnih področij na upravljanje konfiguracij močno vplivajo. Velja seveda tudi obratno.

Problemi, ki so jih uporabniki v anketi najbolj izpostavili, so večinoma povezani z enim od naslednjih področij:

- *upravljanje s produktno linijo in življenjskimi cikli proizvodov.* Najbolj izstopajoča problema sta težavno obvladovanje razvoja številnih različic proizvodov in vodenje življenjskega cikla produkta, kjer ni kontrolne verige skozi celotni cikel;
- *upravljanje zahtev za proizvode.* Obstajajo pomanjkljivosti v vodenju liste zahtev, v sledljivosti po razvojnih fazah in funkcionalnostih pri vnosu sprememb, ter poznem vnašanju sprememb v različne produkte;
- *nadzor nad spremembami v programski opremi* ni optimalen. Nimamo popolnega nadzora nad spremembami, ki v pakete prihajajo do projektnega mejnika M570 (Frey-Pučko, 2002, str. 10), ki pomeni konec sistemske faze verifikacije. Definirani postopki se zaradi omenjenega razloga lahko obidejo po bližnjicah. V programske pakete pozno prihajajo spremembe zaradi odpravljanja napak in novih funkcionalnosti. Problematičen je tudi pojav, ko se s spremembami v skupnih gradnikih pred in po B600 nenadzorovano

vnesejo napake v že stabilne proizvode na terenu. Včasih ostanejo zaradi časovnega pritiska ob izdajanju paketov spremembe pri potrjevanju premalo preverjene (inspekcija);
- *verifikacija*. Izstopata pregled nad testiranostjo produkta (predvsem v fazi implementacije) in povezava rezultatov testiranj s konfiguracijo izdelkov;
- *planiranje tehničnih aktivnosti.* Problematičen je pregled nad tekočim stanjem planiranja aktivnosti. Težavno je določanje in usklajevanje prioritet za aktivnosti;
- *nadzor nad izvajanjem tehničnih aktivnosti.* Definirani postopki so premalo upoštevani, hkrati je razvojne aktivnosti za številne različice produktov težko nadzorovati.

3.2.4 Identificirani predlogi izboljšav

Med preučevanjem trenutnega problemskega stanja je bilo v preteklem obdobju s strani sodelavcev podanih razmeroma veliko predlogov za izboljšave. Te izboljšave lahko v grobem razdelimo na kratkoročne korektivne ukrepe in dolgoročne celovite izboljšave.
Primeri pomembnejših kratkoročnih korektivnih ukrepov (Frey-Pučko, 2002, str. 14): dosledno upoštevanje že sprejetih pravil; realizacija planiranih povezav med obstoječima sistemoma za SCM in PDM; izboljšava upravljanja sprememb v vseh fazah razvoja, predvsem pri planiranju in odločanju o vključevanju sprememb v proizvode; izboljšan nadzor nad procesom s strani vseh udeleženih.
Iz razlogov, navedenih v prejšnjih razdelkih, je mogoče bistvene, dolgoročnejše izboljšave doseči samo s hkratnimi izboljšavami procesov in orodij. V zvezi s problematiko ostalih procesov, ki so z upravljanjem konfiguracij neposredno povezani, je potrebno najprej postaviti jasna izhodišča za reševanje in koliko odprtosti bomo dopustili. Izboljšav se bo potrebno lotiti vsaj na naslednjih ravneh oziroma področjih:

- vodenje produktne linije;
- upravljanje zgradbe (arhitekture) izdelkov;
- odločitve glede načina vodenja generike in specifik proizvodov;
- načini spreminjanja proizvodov v zreli fazi;
- preučitev in določitev ključnih zahtev za izvajanje vzporednega razvoja in vzdrževanja;
- upravljanje zahtev (za proizvode), ki neprenehoma vletavajo v proces;
- proces planiranja in vodenja projektov;
- proces planiranja in upravljanja konfiguracij in sprememb izdelkov in njihove PO;
- podporni sistemi za vsa zgoraj omenjena področja, zagotovitev ustreznega nivoja avtomatizacije in njihove medsebojne povezanosti.

Pri izbiri orodij za podporo procesu upravljanja konfiguracij je potrebno upoštevati, da imajo vsa komercialna orodja, ki nastopajo v eni od možnih rešitev, opisanih v nadaljevanju, vsaj v nekaterih ključnih funkcionalnostih, kot je npr. upravljanje sprememb, že vgrajen vnaprej določen proces oziroma postopke (Frey-Pučko, 2002, str. 14) - to so t.im najboljše izkušnje oziroma pravila dobre prakse (angl. best practices). Pomembnejša razloga, da so ta vsebovana neposredno v orodju, sta zagotoviti konsistentnost upravljanja konfiguracij in sprememb in zavezati uporabnike k upoštevanju postopkov, brez nenadzorovanih odstopanj. Odločitev za nakup določenega komercialnega orodja tako hkrati pomeni tudi odločitev, da v določenih ključnih segmentih privzamemo proces, ki ga je proizvajalec vgradil v orodje.

4. Izgradnja odločitvenega modela

V tem poglavju je podan opis odločitvenega modela, uporabljenega v procesu ovrednotenja sistemov za podporo upravljanju konfiguracij izdelkov in njihove PO. Pri ocenjevanju se uporabljata dva glavna kriterija: konkurenčna sposobnost sistema in ustrezanje zahtevam podjetja. V nadaljevanju so našteti in predstavljeni kriteriji, ki sestavljajo oba omenjena glavna kriterija. Opisan je način njihove ureditve (strukturiranja), glavne značilnosti uporabljenih funkcij koristnosti, ovrednoten pa je tudi pomen ključnih kriterijev.

4.1 Kriteriji

V tem podpoglavju je zajet opis kriterijev, ki smo jih uporabili pri ovrednotenju sistemov za upravljanje konfiguracij proizvodov in njihove programske opreme. Za razvrščanje ocenjenih sistemov smo uporabili metodo portfelja.

4.1.1 Uporabljeni kriteriji

Za izdelavo portfelja sta bila, kot že omenjeno, uporabljena glavna kriterija: konkurenčna sposobnost sistemov orodij za upravljanje konfiguracij in zahteve podjetja Iskratel (za sistem). Glavna kriterija sta bila določena z vrsto nadaljnih kriterijev.

4.1.1.1 Kriteriji v zvezi s konkurenčno sposobnostjo

Seznam kriterijev:

- *funkcionalnost* - funkcionalnosti sistema in njihova kakovost;
- *konfiguracije* - kakovost rešitev na področju upravljanja konfiguracij in sprememb;
- *projekti* - podpora vodenju in izvajanju projektov;
- *nadzor* - podpora načrtovanju in spremljanju poteka projektnih nalog, aktivnosti;
- *timsko delo* - podpora u razvojnih skupin;
- *veliki projekti* - možnost podpore velikih, kompleksnih projektov;
- *proces* - raven podpore procesa upravljanja kofiguracij in sprememb izdelkov;
- *avtomatizacija* - način in stopnja avtomatizacije izvajanja operacij;
- *vloge* - podpora različnih vlog v procesu;
- *modeliranje* - možnosti za modeliranje procesov;
- *zgradba sistema* - kako ustrezna je zgradba (arhitektura) sistema;
- *trdoživost* - trdoživost, robustnost, zanesljivost, odpornost na napake in okvare;
- *odprtost* - ustrezanje določenim standardom in možnost dodajanja modulov, ki ustrezajo enakim standardom;
- varnost - zagotavljanje varnosti uporabe podatkov;
- *poslovna uspešnost* - poslovna uspešnost proizvajalca in samega sistema;
- *tržna pozicija* - tržna pozicija proizvajalca in sistema;
- *sistem* - uveljavljenost podpornega sistema;
- *proizvajalec* - tržna pozicija in ugled podjetja, ki sistem razvija in vzdržuje;
- *perspektiva* - tehnološka in poslovna perspektiva proizvajalca;
- *podpora* - raven podpore kupcem oziroma uporabnikom sistema;
- *vpeljava* - pomoč pri vpeljavi sistema (npr. svetovanje, šolanja,...);
- *izboljšave* - stalne izboljšave sistema in tekoče reševanje problemov na sistemu.

4.1.1.2 Kriteriji v zvezi z izpolnjevanjem zahtev podjetja

Seznam kriterijev:

- *zahteve sistema* - ustrezanje zahtevam informacijskega sistema podjetja;
- *integracija v sistem* - možnost učinkovite povezave v informacijski sistem podjetja;
- *platforme* - podpora operacijskim sistemom, ki se uporabljajo pri razvoju SI2000;
- *podporni sistemi* - možnost integracije z drugimi sistemi za podporo poslovanja;
- *razvojna okolja* - podpora obstoječih razvojnih okolij in aplikacij;
- *vmesniki* - ustreznost uporabniških vmesnikov;
- *intuitivnost* - prijaznost, intuitivnost, učinkovitost uporabniških vmesnikov;
- *kompletnost* - možnosti raznovrstne uporabe uporabniških vmesnikov;
- *posebne zahteve* - posebne zahteve procesa in metodologij;
- *modularnost* - podpora delu z moduli in komponentami proizvodov;
- *zahteve* - možnost učinkovitega upravljanja z zahtevami za proizvode SI2000;
- *distribuirani razvoj* - podpora razvoja izdelkov na geografsko ločenih lokacijah;
- *zadovoljstvo* - ocena potencialnega zadovoljstva ožjega in širšega kroga uporabnikov;
- *ugodnost investicije* - kako ugodna je naložba za podjetje;
- *višina investicije* - skupen obseg naložbe v podporni sistem;
- *donosnost* - ocena razmerja med pričakovanim donosom in vloženimi sredstvi - za obdobje petih let (kazalnik ROI);
- *ugodnosti* - finančne in druge ugodnosti, ki jih nudi proizvajalec ali dobavitelj sistema;
- *celovitost rešitve* - celovitost rešitve v smislu izpolnitve namena in dolgoročnosti;
- *pokritost* - pokritost funkcionalnosti upravljanja konfiguracij in sprememb izdelkov;
- *uporabnost* - dolgoročnost uporabnosti rešitve (brez večjih posegov);
- *partnerstvo* - pričakovana kakovost sodelovanja s proizvajalcem in/ali dobaviteljem;
- *prehod* - ocena enostavnosti prehoda na novo podporno okolje;
- *kompleksnost sistema* - težavnost uporabe sistema za različne uporabnike;
- *prilagodljivost* - prilagodljivost sistema;
- *skladnost procesa* - skladnost z zahtevami razvojnega procesa in procesa upravljanja konfiguracij.

4.1.2 Kratek opis kriterijev

4.1.2.1 Kriteriji v zvezi s konkurenčno sposobnostjo

Konkurenčna sposobnost. Ponazarja raven konkurenčne sposobnosti sistemov za upravljanje konfiguracij proizvodov. Odvisna je predvsem od: funkcionalnosti sistema in njihove kakovosti; zgradbe oziroma zasnove sistema ter poslovnih dejavnikov, npr. tržne pozicije sistema in njegovega proizvajalca sistema ter podpore, ki jo proizvajalec in/ali dobavitelj sistema orodij za upravljanje konfiguracij nudita kupcu oziroma uporabnikom.

Funkcionalnost. Ocena kakovosti pokrivanja posameznih področij upravljanja konfiguracij in sprememb. Sistem mora na ustrezen način pokrivati upravljanje konfiguracij in sprememb proizvodov in njihove PO, dobro mora podpirati vodenje in izvajanje projektov, poleg tega pa mora na ustreznem nivoju podpirati izvajanje tako razvojnega in vzdrževalnega procesa kot procesa upravljanja konfiguracij in sprememb proizvodov in njihove PO.

Konfiguracije. Od sistema pričakujemo kakovostno upravljanje konfiguracij izdelkov in njihove PO, prav tako pa tudi kakovostno upravljanje sprememb nad gradniki na obeh navedenih ravneh. Od sistema se v zvezi z upravljanjem konfiguracij pričakuje, da bo na ustrezni ravni

pokrival področje nadzora verzij/revizij različnih zvrsti gradnikov: tekstovnih datotek, binarnih datotek in ostalih tipov gradnikov (testni vzorci, knjižnice itd.).
V sklopu tega kriterija naj bi sistem orodij na ustrezen način podpiral gradnjo in izdajanje PO. Pričakujejo se naslednje lastnosti: usklajenost z orodjem make; ponovljivost procesa izgradnje paketov PO; izogibanje gradnje že zgrajene PO s pouporabo izpeljanih objektov; sledenje in razpoznavanje povezav med gradniki itd.
V zvezi z upravljanjem sprememb se pričakuje podpora življenjskega cikla sprememb, od javljanja napake oziroma pripetljaja na proizvodu in podajanja zahteve za spremembo, do realizacije in testiranja ustreznih sprememb ter njihovi izdaji v novi verziji proizvoda.
Vse omenjene operacije bi morale delovati učinkovito v kontekstu sočasnega dela na družinah sorodnih proizvodov.
Projekti. Tu je zajeta podpora vodenja in izvajanja projektov z vidika upravljanja konfiguracij in sprememb nad proizvodi. Sistem naj bi zagotavljal sprejemljivo podporo pri spremljanju in nadzoru projektov, predvsem na ravni planiranja, poverjanja in spremljanja nalog v zvezi s spreminjanjem komponent izdelkov. Sistem mora ponuditi tudi ustrezno podporo skupinskemu načinu dela in zagotavljati podporo velikih, kompleksnih projektov.
Nadzor. Zaželjeno je, da ima sistem sprejemljive možnosti upravljanja s človeškimi in materialnimi viri pri izvajanju razvojnih nalog. Manj pomembna je možnost planiranja na nivoju celotnih projektov, njihovih ključnih faz in časovnih vložkov zanje (v ta namen se uporabljajo druga orodja). Od sistema se pričakuje, da bo omogočal izdelavo in razpošiljanje kakovostnih poročil v zvezi s trenutnim stanjem aktivnosti oziroma nalog na projektu, predvsem v povezavi s tekočim stanjem konfiguracije izdelkov in njihovih komponent.
Timsko delo. Od orodij pričakujemo podporo timskemu delu, npr. s podporo delu skupine razvijalcev nad skupno množico gradnikov. To se nanaša na naslednje funkcije sistema:

- učinkovite mehanizme za doseganje gradnikov v odlagališču podatkov, omogočanje enostavnih in učinkovitih mehanizmov za odjavo/prijavo gradnikov v odlagališču itd.;
- podpora vejitev v življenjskih ciklih razvoja gradnikov in komponent izdelkov, vključno z operacijo združevanja (angl. merge) različnih verzij gradnikov;
- omogočanje učinkovitih primerjav vsebin različnih verzij gradnikov; itd.

Veliki projekti. Pričakuje se podpora procesu upravljanja konfiguracij in sprememb nad proizvodi za velike in kompleksne projekte. Ne sme se zgoditi, da bi zaradi rasti projekta (npr. v številu gradnikov, številu uporabnikov v projektu, v večanju obsega komponent in števila povezav med njimi) prihajalo do problemov pri uporabi sistema.
Proces. Od sistema pričakujemo, da v sprejemljivi meri podpira proces upravljanja konfiguracij izdelkov, prav tako pa tudi proces upravljanja sprememb izdelkov. Pri tem kriteriju je pomembno, kakšne so možnosti podpore različnih življenjskih ciklov (npr. življenjskega cikla gradnika, komponente, celotnega izdelka, spremembe na izdelku, posamezne naloge...). V tem smislu je zaželjena možnost modeliranja raznih tipov procesov in življenjskih ciklov.
Uspešnost podpore procesa je odvisna tudi od drugih dejavnikov, npr. na kakšen način je proces avtomatiziran (s sprožilci, z obvladovanjem družin procesov ipd.), ali orodja na ustrezen način podpirajo različne vloge v razvojnem procesu ipd.
Avtomatizacija. Od sistema orodij za upravljanje konfiguracij izdelkov in njihove PO se pričakuje, da na ustrezni ravni podpira predvidene funkcije. Pri tem je pomembno, da so vsi koraki razvojnega procesa podprti čim bolj enakomerno, tako da v procesu ni ozkih grl.
V zvezi s kakovostjo podpore procesa je pomembno, na kakšen način je implementiran nadzor procesa, npr.: nadzor procesa sploh ni predviden; na osnovnem nivoju je podprt z nekaterimi orodji; predviden je s sprožilci; orodja za SCM podpirajo oblikovanje in nadzorovanje družin procesov (npr. z objektnim pristopom), itd.

Vloge. Od sistema pričakujemo, da podpira ustrezne akcije in postopke za različne vloge v procesu upravljanja in konfiguracij sprememb, kot npr. razvijalec, vzdrževalec, vodja projekta, avtor gradnika, avtor spremembe, odgovorni za izvedbo integracije ipd.

Modeliranje. Sistem naj bi imel solidno podlago za upravljanje in uporabo procesnih modelov. Pomemben vidik tega kriterija predstavlja možnost enostavnega prilagajanja teh modelov, npr. morebitnim dodatnim lastnim zahtevam.

Zgradba. Zgradba (arhitektura) sistema je poleg funkcionalnosti sistema en ključnih dejavnikov, ki vplivajo na konkurenčno sposobnost sistema. Sistem mora biti trdoživ (robusten), omogočati mora nemoteno delo tudi v primeru pojavljanja raznih težav. Poleg tega mora biti sistem odprt v smislu dograjevanja z drugimi funkcionalnimi moduli. Zasnovan mora biti tako, da zagotavlja varen način hranjenja in obdelave podatkov.

Trdoživost. Da bo sistem konkurenčen, mora imeti trdoživo (robustno) zgradbo, ki bo zagotavljala ohranjanje konsistentnosti vseh vodenih podatkov o izdelkih in nemoteno uporabo sistema tudi v primeru morebitne nepravilne uporabe ali motenj v informacijski infrastrukturi (npr. v računalniški mreži). Ta kriterij mora biti izpolnjen ne glede na to, ali se sistem uporablja v lokalni računalniški mreži ali v geografsko porazdeljenem razvojnem sistemu.

Odprtost. Sistem mora imeti kar se da odprto zgradbo. Zgradba naj bi ustrezala določenim pravilom oziroma standardom, kar pomeni, da obstaja možnost dodajanja drugih modulov ali povezovanja z drugimi sistemi za podporo poslovanju, ki ustrezajo enakim standardom. Ta dejavnik v sodobnem poslovnem okolju, kjer v vse večji meri prevladujejo geografsko porazdeljeni poslovni sistemi, dobiva na veljavi. Vse pomembneje je, da podatki niso več le neizkoriščen kapital v odlagališčih poslovnih aplikacij, temveč da jih je mogoče kar se da učinkovito dosegati in uporabljati tudi iz drugih poslovnih sistemov in okolij.

Varnost podatkov. Sistem mora biti zasnovan in zgrajen tako, da bo omogočal varnost hranjenja in uporabe podatkov, tako v primeru raznih nepričakovanih nezgod kot v primeru morebitnih poskusov nepooblaščene uporabe oziroma zlorabe podatkov.

Poslovna uspešnost. Tu sta zajeta tržna pozicija podpornega sistema in njegovega proizvajalca ter raven podpore uvajanju in uporabi sistema s strani dobavitelja oziroma proizvajalca sistema.

Tržna pozicija. V ta kriterij sta zajeta tako uveljavljenost sistema kot podjetja oziroma podjetij, ki razvijajo sistem ali njegove posamezne komponente. Pomembna je tako tehnološka kot poslovna perspektiva. Primer kriterija, na podlagi katerega lahko ocenimo perspektivo, je, ali podjetje snuje nove napredne pristope (npr. nove procesne modele, vključno z njihovo avtomatizirano podporo) na področju upravljanja konfiguracij izdelkov in njihovih komponent.

Sistem. Uveljavljenost oziroma tržna pozicija sistema pove precej o samem sistemu. Pomembna informacija s tem v zvezi je lahko intenzivnost rasti oziroma spreminjanja tržnega deleža danega sistema. Dejstvo, da mnoga razvojno usmerjena podjetja kupujejo, uvajajo ali že uporabljajo tak sistem, povečuje verjetnost glede resničnosti trditve, da se nakup in uporaba danega sistema izplača. Poleg tega velja tudi, da dve partnerski podjetji pri razvoju izdelka lažje in učinkoviteje sodelujeta, v kolikor za upravljanje konfiguracij tega proizvoda in njegovih komponent uporabljata enaka ali povezljiva podporna sistema.

Proizvajalec. Od podjetja oziroma podjetij, ki proizvajajo ali dobavljajo kakovostne sisteme (ali njihove komponente) za upravljanje konfiguracij izdelkov pričakujemo, da bodo ali že uveljavljena na tržišču, z ustreznimi referencami med uporabniki, ali da bo njihov tržni delež iz leta v leto v ustrezni meri naraščal.

Perspektiva. Tu sta mišljeni tako tehnološka kot z njo povezana poslovna perspektiva proizvajalca sistema oziroma proizvajalcev ključnih komponent sistema. Od proizvajalcev sistemov pričakujemo, da bodo razvijali nove uspešne in perspektivne pristope na področju

upravljanju konfiguracij in nadzora sprememb proizvodov, tako z vidika različnih funkcionalnosti sistema in njegovih zmogljivosti, kot tudi z vidika obvladovanja ustreznih procesov. Pomembno je tudi, kako je podjetje sposobno svoje rešitve tržiti in si zagotoviti ustrezno pozicijo na trgu.

Podpora. Pomembno je, da podjetje, ki razvija in/ali dobavlja podporne sisteme oziroma njihove ključne module, sistema ne ne le prodaja, ampak tudi: pomaga pri uvajanju sistema v podjetju, ki sistem vpeljuje; ponuja dodatno izobraževanje v zvezi z administriranjem in uporabo sistema; da se pravočasno in korektno odziva na napake in pomanjkljivosti, najdene pri uporabi orodij ter upošteva koristne predloge pri nadaljnjih izboljšavah svojega sistema (ali modula) za upravljanje konfiguracij izdelkov, če in ko je to potrebno.

Vpeljava. Ali podjetje oziroma podjetja, ki razvijajo sistem ali njegove ključne module, nudijo kakšno podporo pri uvajanju v organizacijah, ki kupijo ta orodja (svetovanje ipd.)? Ali dobavitelji oziroma izdelovalci sistema ponujajo ustrezno dodatno izobraževanje v zvezi z administriranjem in uporabo sistema oziroma njegovih ključnih modulov?

Izboljšave. Ali podjetja, ki razvijajo in/ali dobavljajo ključne komponente sistema, upoštevajo javljene pripombe v zvezi z odkritimi napakami oziroma pomanjkljivostmi in jih upoštevajo pri nadaljnjem razvoju sistema? Ali so nove verzije sistema boljše v primerjavi s prejšnjimi, ali ponujajo nove, učinkovitejše, kakovostnejše tehnološke rešitve?

4.1.2.2 Kriteriji za izpolnjevanje zahtev podjetja

Zahteve podjetja. Ta dejavnik ponazarja, v kolikšni meri je podporni sistem sposoben zadovoljevati poslovne zahteve ter trenutne in prihodnje zahteve procesa razvoja in vzdrževanja proizvodov SI2000 in njihove PO. Pri tem so pomembni naslednji dejavniki: ustrezanje zahtevam razvojnega in podpornega sistema, metodologij, procesa; zadovoljstvo vodstva in ključnih uporabnikov sistema; enostavnost prehoda na novo okolje.

Zahteve sistema. Pričakujemo, da bo dani sistem mogoče učinkovito povezati z obstoječimi sistemi in razvojnimi metodologijami ter procesi.

Sistem naj bi imel tudi ustrezne uporabniške vmesnike. Le-ti morajo biti intuitivni in prijazni do uporabnika, imeti čim bolj enotno zunanjo podobo ter imeti dovolj možnosti za vsestransko uporabo (uporaba tako vrstičnih kot grafičnih vmesnikov).

Poleg tega mora sistem orodij ustrezati vsem pomembnejšim posebnim zahtevam procesa in metodologij pri upravljanju konfiguracij in sprememb izdelkov SI2000 (npr. podpora delu na geografsko ločenih razvojnih lokacijah, podpora modularnosti ipd.).

Integracija v sistem. Dejavnik ponazarja oceno možnosti integracije v razvojni in podporni sistem, kamor spadajo: informacijska infrastruktura, kjer prevladujeta operacijska sistema Unix in Windows (v večih izvedbah in različicah); aplikacije in okolja za razvoj različnih komponent proizvodov; sistemi za podporo drugih vidikov poslovanja, povezanih z življenjskim ciklom proizvodov SI2000. Zaželjeno bi bilo, da bi sistem deloval obeh glavnih razvojnih platformah v podjetju, tako na HP-UX kot na Windows NT. Poleg tega bi bilo zelo ugodno, če bi uporabniki lahko delali na sistemu tudi preko Web odjemalcev. Čim bolj učinkovita bi bila raven povezav sistema na navedenih ravneh, tem bolje bi bile izkoriščene zmožnosti sistema.

Platforme. V tem kriteriju je zajeta pokritost glavnih razvojnih platform (Windows, HP-UX) in Web okolju. V podjetju se poleg navedenih dveh razvojnih platform uporabljajo tudi druge, vendar so te manj pomembne (npr. SCO Unix). V Zadnjem času je vse bolj zanimiv Linux.

Na sistemih Windows in HP-UX teče cela vrsta razvojnih okolij: SDL Geode, Tornado, pSOS, Java... To sta široko uporabljani računalniški platformi v mnogih konkurenčnih podjetjih. Od sistema za upravljanje konfiguracij in sprememb pričakujemo, da bo kar se da dobro podprl delo na obeh. Podpora odjemalcev za Web je pomembna predvsem zaradi lažjega, enostavnejšega dostopanja do ključnih storitev in podatkov o izdelkih.

Podporni sistemi. Od sistema pričakujemo, da bi imel možnost povezovanja z drugimi sistemi za podporo poslovanja, ki se že uporabljajo v našem poslovnem sistemu, posebej s tistimi, ki se nanašajo na izdelke SI2000 (Primavera, Remedy, mySAP.com na področju financ, logistike in proizvodnje ipd.). V tem trenutku bi bili zadovoljni, v kolikor bi imeli zagotovljene neposredne povezave preko vmesnikov, vgrajenih v sama orodja. Pričakujemo, da bo v prihodnosti v ta namen vse bolj aktualna uporaba povezovalne programske opreme in standardov za povezovanje aplikacij in poslovnih sistemov.

Razvojna okolja. Od orodij za upravljanje konfiguracij in sprememb se pričakuje, da so na določeni ravni sposobna podpreti različna razvojna orodja in aplikacije, ki se uporabljajo v razvojnem procesu izdelkov SI2000: Telelogic Geode, Tornado, Microsoft Developer Studio, NewEra, Java, SDRC IDEAS, Menor Graphics ipd.

Vmesniki. Od sistema se pričakuje ustrezna kakovost uporabniških vmesnikov. Po naših izkušnjah so lahko slabi, t.j. nefunkcionalni, nepregledni, togi in estetsko neprimerni uporabniški vmesniki vir za nezadovoljstvo in odpor uporabnikov do uporabe podpornih sistemov. Pomembno je, da so uporabniški vmesniki prijazni in intuitivni. S tem ni omogočeno le večje zadovoljstvo, temveč tudi višja učinkovitost uporabnikov sistema.

Zaželjeno je, da obstajajo tako grafični kot ukazno-vrstični vmesniki za dane operacije. Z grafičnimi vmesniki je interaktivno delo uporabnikov lažje in učinkovitejše, z ukazno-vrstičnimi vmesniki pa je omogočena večja prilagodljivost uporabe orodja (npr. v smislu prilagajanja sistema in uporabe operacij sistema v programih, skriptih...).

Prijaznost. Dejavnik, ki ponazarja, kolikšna je stopnja prijaznosti uporabniški vmesnikov (ali so pregledni, intuitivni, enostavni in učinkoviti za uporabo, ali upoštevajo standarde oblikovanja ipd.).

Kompletnost (uporabniških vmesnikov). Ali imajo uporabniki na razpolago več tipov vmesnikov (izvajanje ukazov iz ukazne vrstice, izvajanje preko grafičnih vmesnikov ipd.), ali pa lahko uporabljajo le eno vrsto vmesnikov. V splošnem je zaželjena čim večja izbira uporabniških vmesnikov, saj ima vsaka zvrst vmesnikov svoje prednosti. Kljub tej izbiri pa je potrebno paziti tudi na kar se da enotno zunanjo podobo le-teh.

Posebne zahteve. Tu so mišljene specifične zahteve z ozirom na način razvoja telekomunikacijskih izdelkov, kot obstaja v podjetju. Te zahteve so aktualne že danes, pričakovati pa je, da bo njihova pomembnost v prihodnosti le še naraščala:

- podpora konceptu modularnosti ("modulov") pri upravljanju izdelkov SI2000;
- podpora učinkovitemu upravljanju zahtev za izdelke;
- podpora razvoju proizvodov na različnih geografskih lokacijah. Ta zahteva je vse bolj aktualna, saj poteka razvoj izdelkov SI2000 na različnih lokacijah v Sloveniji in tujini.

Modularnost. Zelo bi bilo zaželjeno, da bi sistem orodij na ustrezen način podpiral modularnost oziroma učinkovito delo s komponentami (npr. "funkcionalnimi moduli", t.j. funkcionalno zaokroženimi množicami gradnikov) proizvodov SI2000. S tem bi ohranili oziroma nadgradili mehanizme za upravljanje zgradbe izdelkov in zagotovili pouporabljivost in lažje možnosti vzdrževanja teh komponent, po drugi strani pa bi to vplivalo na večjo učinkovitost pri razvoju in vzdrževanju družin sorodnih izdelkov.

Zahteve. Proces upravljanja zahtev za proizvode SI2000 trenutno ni niti najbolje definiran niti najbolje podprt in avtomatiziran z računalniškimi orodji. Zato bi bilo zaželjeno, da podporni sistem ponuja nekaj rešitev tudi za to področje, npr. preko že omenjenih vgrajenih najboljših izkušenj oziroma pravil dobre prakse ("best practices"). Zaželjeno bi bilo, da bi sistem dajal dober pregled nad tekočim stanjem posamezne zahteve za proizvod, od faze sprejema zahteve in analize, do realizacije, testiranja in ne nazadnje preverjanja kupčevega zadovoljstva.

Distribuirani razvoj. Ali sistem orodij predvideva rešitve za podporo procesa razvoja in vzdrževanja izdelkov v razvojnih centrih na različnih geografskih lokacijah. V podjetju se že pojavlja določen obseg zahtev po tovrstni podpori, vendar jih trenutno uspešno obvladujemo z lastnim sistemom, ki pa ima nekaj izvedbenih omejitev. Od sistema pričakujemo ustrezne možnosti za avtomatizirano podporo geografsko porazdeljenemu razvoju oziroma možnost, nadgraditi sistem brez posebnih dodatnih težav (npr. s posebnim dodatnim modulom).

Zadovoljstvo. Pred vpeljavo sistema za podporo upravljanju konfiguracij proizvodov se je potrebno vprašati, v kolikšni meri bi bili s tem sistemom zadovoljni tako bodoči uporabniki (razvijalci in vzdrževalci izdelkov SI2000, izvajalci servisne podpore izdelkov SI2000, vodje in koordinatorji razvojnih projektov, vodje produktov...) kot nenazadnje vodstvo poslovne enote. Pri sistemu, s katerim bi bili uporabniki nezadovoljni, bi morali, v skladu z izkušnjami iz preteklih tovrstnih projektov, računati na dodatne stroške v zvezi z učinkovitostjo tako vpeljevanja sistema v uporabo kot same uporabe. Gre za sociološki, kulturni problem. Dejavniki, od katerih je odvisno zadovoljstvo vodstva poslovne enote in ključnih skupin uporabnikov navedenega podpornega sistema v našem poslovnem okolju, so:

- ugodnost naložbe (v največji meri zanima vodstvo poslovne enote);
- kako celovita bo sprejeta rešitev, tako v smislu izpolnitve namena kot glede dolgoročnosti (zanimivo za vodstvo in širok krog uporabnikov);
- pričakovana kakovost sodelovanja s proizvajalci oziroma dobavitelji sistema (ali ključnih modulov sistema), tako v fazi nakupa in vpeljave kot pozneje, med samo uporabo in vzdrževanjem sistema, je dejavnik, ki vpliva na zadovoljstvo skupine sistemskih inženirjev, zadolženih za administriranje sistema, izvajanje prilagoditev sistema ipd.

Ugodnost investicije. Tu je mišljena ugodnost naložbe, glede na njeno višino (povezano s trenutno investicijsko sposobnost podjetja), prav tako pa tudi z ozirom na koristi, ki bi nam jih prinesel morebitni nakup določenega sistema v toku njegove uporabe. V primeru obsežne naložbe je pomembno, ali obstajajo kakšne (npr. finančne) ugodnosti pri nakupu sistema.

Višina investicije. Obseg naložbe v nakup, vpeljavo, uporabo in vzdrževanje sistema (merjeno v ameriških dolarjih).

Donosnost. Ocena kazalnika ROI - t.j. količnik, ki pove, koliko denarnih enot dobimo v zameno za eno denarno enoto, ki smo jo vložili v nov sistem oziroma v projekt vpeljave sistema. Vrednost denarja se pri tem izračunu praviloma prevede na njegovo sedanjo vrednost. Pri izračunu je zajeto omejeno obdobje (npr. pričakovana življenjska doba uporabe podpornega sistema) - v našem primeru smo vzeli obdobje petih let. Pričakujemo, da bo sistem v tem obdobju ustrezno izpolnjeval svoj namen in da bo skupna korist, ki jo bomo v tem obdobju deležni od uporabe sistema, večja od vložkov v celotno investicijo.

V nakup sistema bi bi se torej praviloma podali le pod pogojem, da bi nam ta v proces prinesel korist, dodano vrednost. Pri izračunu kazalnika je najlažje izmeriti in upoštevati kratkoročne in neposredne koristi, npr. prihranke zaradi povečane učinkovitosti uporabnikov sistema (t.j. predvsem razvijalcev, vzdrževalcev, serviserjev izdelkov itd.). Marsikatero korist, ki jo prinese dober sistem za upravljanje konfiguracij, pa je le težko ustrezno opredmetiti, mnoge imajo tudi dolgoročnejši značaj - npr. višji nivo usklajenosti in povezanosti celotnega sistema vodenja proizvodov, višja kakovost končnega izdelka, posledično višje zaupanje kupcev itd. Ker običajno nismo sposobni v celoti in na ustrezen način oceniti dolgoročnejših in posrednih učinkov vpeljave sistemov za upravljanje konfiguracij v proces, pogosto pa upoštevamo le kratkoročne in neposredne učinke, pri ocenjevanju kazalnika ROI obstaja nevarnost, da dobimo pesimistične ocene.

Celovitost rešitve. Sistem naj ponuja kar se da celovito podporo funkcij upravljanja konfiguracij in sprememb ter vseh ključnih 'robnih funkcij in procesnih področij', na katera mejita omenjena

procesa. Pri tem je pomembno, da s sistemom dobimo rešitev, ki bo brez večjih dodatnih posegov služila svojemu namenu v naslednjem obdobju (npr. petih let).

Pokritost področja upravljanja konfiguracij in sprememb. V preteklosti smo imeli nekaj sistemov, ki so le delno reševali zahteve obeh procesov oziroma ravni upravljanja, s čimer niti vodje niti izvajalci projektov niso bili posebej zadovoljni. Izkušnje so pokazale, da več kot je sistemov na nekem področju, kompleksnejši je sistem, več je 'razpok', 'šivov' v sistemu in bolj se delo podvaja.

Uporabnost rešitve. Od danega sistema pričakujemo, da bo zadovoljil naše zahteve za naslednjih nekaj let. Pri tem predpostavljamo, da se bodo sistemi na področju upravljanja konfiguracij in sprememb razvijala v bodoče s podobno intenzivnostjo kot do zdaj.

Partnerstvo. Kriterij ponazarja pričakovano kakovost sodelovanja s proizvajalci in dobavitelji, predvsem na podlagi dosedanjih izkušenj z njimi, pa tudi na podlagi drugih informacij. Od proizvajalca in/ali dobavitelja sistema pričakujemo resen pristop k projektu vpeljave novega sistema v uporabo. Upoštevati je potrebno izkušnje iz morebitnih prejšnjih skupnih projektov. V kolikor neposrednih izkušenj nimamo, upoštevamo informacije, ki jih v zvezi s tem dobimo od naših poslovnih partnerjev ali drugih podjetij, ki imajo tovrstne izkušnje.

Pomembno je npr., kako proizvajalci in/ali dobavitelji predstavljajo svoje izdelke oziroma module in koliko 'pravih', kakovostnih informacij so pripravljeni posredovati ter v kolikšni meri so pripravljeni sodelovati v pripravi morebitnih pilotskih projektov z namenom praktičnega preizkušanja in ovrednotenja sistema. Od predstavnikov teh podjetij pred nakupom sistema pričakujemo sodelovanje pri pripravi ustreznih rešitev v zvezi z uporabo sistema. Proces odločanja pri tovrstnih projektih je lahko dolgotrajen, že v samo ovrednotenje sistemov pa je običajno vloženo precej finančnih sredstev in časa sodelujočih.

Prehod. Ta kriterij predstavlja oceno težavnosti prehoda na novo podporno okolje. Po izkušnjah iz sorodih projektov je pomembno, da se uporabniki že takoj ob začetku dela na sistemu ne soočijo z nepremostljivimi ovirami, saj to vpliva ne le na njihovo mnenje o sistemu, temveč tudi na učinkovitost njihovega dela. Kriterij je neposredno povezan z dejavniki, kot so kompleksnost in prilagodljivost sistema ter stopnja skladnosti med procesom, ki se trenutno izvaja v podjetju, in procesom, ki je vgrajen v sistem za upravljanje konfiguracij.

Kompleksnost sistema. Dejavnik ponazarja, kako zapleten je sistem za administriranje in uporabo. Od sistemov za podporo upravljanja konfiguracij izdelkov se pričakuje, naj ne bi bili preveč zapleteni, saj je dovolj kompleksen že sam proces, ki ga podpirajo. Od njih se pričakuje ravno obratno - poenostavitev in avtomatizacijo tega procesa. Pomembno je, da zapletenost ni prevelika niti za administratorje sistema niti za njegove uporabnike (razvijalci, vzdrževalci ipd.).

Prilagodljivost. Ali je sistem orodij za upravljanje konfiguracij in sprememb mogoče prilagoditi razvojnemu procesu, okolju, metodologijam (tako na ravni celotnega sistema kot na ravni dane razvojne skupine ali celo posameznega uporabnika) ali pa so orodja za upravljanje konfiguracij in sprememb toga. V tem smislu pričakujemo ustrezno prilagodljivost sistema, kar pomeni, da bi morebitne naknadne prilagoditve lahko izvedli na enostaven in učinkovit način. Večja kot je togost sistema, večji je napor (vložek) za izvedbo določene spremembe na sistemu.

Po drugi strani velika prilagodljivost podpornega sistema ni nujno prednost. Za to obstajajo vsaj trije razlogi. Prvi je ta, da premalo premišljen način prilagajanja sistema sicer lahko koristi določeni skupini uporabnikov, ki je dovolj glasna, da vsili svojo zahtevo, pri tem pa lahko na negativen način vpliva na delo drugih uporabnikov, ki pa se 'tiho' sprijaznijo s spremembo. Drug razlog je v tem, da prepogosto spreminjanje ali postopkov v zvezi z uporabo sistema ali podatkovnih struktur in drugih mehanizmov, na katerih sistem temelji, samo po sebi bega uporabnike in onemogoča optimalno učinkovitost in standardiziran način dela. Tretji in zelo pomemben razlog, ki je posebej pomemben pri uporabi komercialnih CM sistemov je v tem, da

je te prilagoditve potrebno vzdrževati, kar za podjetje predstavlja dodaten strošek. Primer: pri vsaki nadgradnji podpornega sistema z novo verzijo je potrebno vse v preteklosti izvedene prilagoditve ponovno preveriti in jih ponovno prilagoditi za delo z novo verzijo sistema.

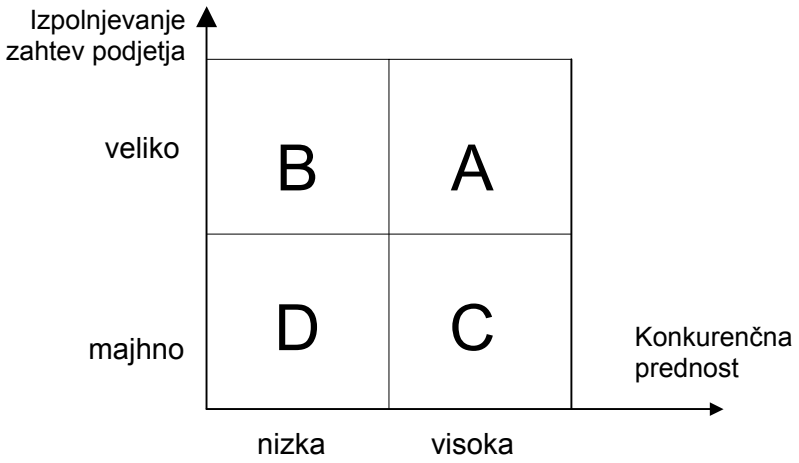
Skladnost procesa. Od sistema pričakujemo, da bo imel v sebi implementiran proces, ki bo ustrezal osnovnim zahtevam v zvezi z obstoječim načinom vodenja izdelkov SI2000 skozi njihov življenjski cikel ter načinom upravljanja njihovih konfiguracij. Poleg tega pričakujemo, da bo ponudil rešitve v smeri pravil dobre prakse (angl. best practices). Pri tem mora biti sistem sposoben zagotoviti možnost razvoja in vzdrževanja družine podobnih telekomunikacijskih izdelkov. V zvezi s tem je zelo pomembno, da sistem nudi zelo dober pregled nad trenutnim stanjem različic omenjenih izdelkov in nad stanjem nalog, povezanih s temi različicami.

V splošnem velja, da večje kot so neskladnosti z osnovnimi zahtevami našega procesa, težji bo prehod na uporabo novega sistema. Pri tem se je potrebno zavedati, da obstoječi proces upravljanja konfiguracij na določenih področjih ni optimalen. Zaželjeno je, da bi bil proces, ki ga bomo uporabljali v prihodnje, še bolj kakovosten. V skladu s tem se v podjetju vseskozi izvaja modeliranje zahtev za optimalno izvajanje razvojnega in vzdrževalnega procesa in njegovih ključnih podprocesov.

4.1.3 Ureditev kriterijev

Ocenjene variante bomo razvrstili v matriki portfelja (slika 18). Kot rečeno, je ta ponazorjena z naslednjima kriterijema: konkurenčna prednost sistemov in izpolnjevanje zahtev podjetja.

Slika 18: Matrika portfelja



Vsak izmed ocenjevanih sistemov se bo uvrstil v neko od navedenih področij matrike: v *področje A* - v primeru visoke konkurenčne sposobnosti in visoke ravni izpolnjevanja zahtev podjetja; v *področje B* - v primeru nizke konkurenčne sposobnosti in visokega izpolnjevanja zahtev podjetja; v *področje C* - v primeru visoke konkurenčne sposobnosti in nizke ravni izpolnjevanja zahtev podjetja; v *področje D* - v primeru nizke konkurenčne sposobnosti in nizkega izpolnjevanja zahtev podjetja. V nadaljevanju so nekoliko podrobneje okarakterizirana področja A, B, C in D matrike portfelja (pri tem se delno zgledujemo po Bostonski matriki):

- v področju A so “zvezde”, zelo kakovostni sistemi z bogatim naborom funkcionalnosti, napredno zasnovo in dobrimi možnostmi medsebojnega povezovanja, ki hkrati dobro ustrezajo zahtevam našega podjetja. O morebitnih kandidatih-zvezdah bi bilo koristno razmisliti in izbrati najprimernejšega izmed njih za preizkus v pilotskem projektu;
- v področju B so “problematični sistemi”. Zanje je značilno, da njihove lastnosti sicer v veliki meri ustrezajo zahtevam našega procesa, vendar po drugi strani njihov nabor

funkcionalnosti ali njihova kakovost ta trenutek (še) ni najboljša. Za problematične sisteme bi v načelu lahko veljala naslednja tri pravila: v kolikor se konkurenčna sposobnost sistema izboljšuje, bi lahko ta orodja kdaj kasneje prišla v poštev za nakup - vendar le v primeru, če se za nakup ne bi odločili v kratkem; v kolikor je sistem orodij prilagodljiv in ima ustrezno ceno ter ustrezno podporo, bi bilo koristno dodatno razmisliti o uporabi teh orodij v podpori našega procesa;

- v področju C so orodja, ki so sicer kakovostna, vendar ne ustrezajo zahtevam podjetja. O nakupu takšnih orodij bi lahko razmišljali le v primeru, če imajo ta orodja velike možnosti za prilagajanje in pri tem njihova cena ni previsoka. V tem primeru bi se jih lahko prilagodilo našim zahtevam. Poleg tega bi bilo potrebno, da za ta orodja obstajajo sprejemljive možnosti za integracijo v naš razvojni proces;
- v področju D so sistemi, ki poleg tega, da ne nudijo najkakovostnejših rešitev na področju upravljanja izdelkov in njihove PO, tudi ne ustrezajo zahtevam našega razvojnega sistema. Ker bi imeli z njimi v prihodnje več stroškov kot koristi, njihova uvedba v naslednjem nekajletnem obdobju ni aktualna.

4.2 Funkcije koristnosti

4.2.1 Glavne značilnosti funkcij koristnosti

V naslednjih dveh razdelkih so podane glavne značilnosti funkcij koristnosti, tako v zvezi s konkurenčno prednostjo kot v zvezi z zahtevami podjetja.

4.2.1.1 Značilnosti funkcij koristnosti za dejavnike konkurenčne sposobnosti

V nadaljevanju so opisane glavne značilnosti funkcij koristnosti za posamezne dejavnike v zvezi s konkurenčno sposobnostjo obravnavanih podpornih sistemov:

- *konkurenčna sposobnost.* V kolikor je ali zgradba ali funkcionalnost sistema neustrezna, je konkurenčna sposobnost ocenjena kot zelo nizka. Oba omenjena dejavnika sta približno enako pomembna oziroma s podobno težo vplivata na končno oceno ter sta nekoliko pomembnejša od tretjega kriterija, t.j. poslovne uspešnosti;
- *funkcionalnost.* Funkcionalnost sistema je ocenjena kot neustrezna, v kolikor je en od osnovnih dejavnikov (t.j. funkcije upravljanja kofiguracij in sprememb izdelkov in njihove PO, zahteve procesa, zahteve v zvezi z vodenjem aktivnosti oziroma nalog v projektih) ocenjen kot slab ali neustrezen. Pri končni oceni nosi največjo težo dejavnik v zvezi s funkcijami upravljanja konfiguracij in sprememb, ostala dva dejavnika pa sta za odtenek manj pomembna, vendar ne nepomembna;
- *projekti.* V kolikor je kateri koli od ključnih dejavnikov ocenjen kot slab ali neustrezno izpolnjen, je izpolnjevanje zahtev v zvezi z vodenjem projektnih aktivnosti ocenjeno kot slabo oziroma nesprejemljivo. Zahteve glede vodenja projektnih nalog so zelo dobro izpolnjenjene, v kolikor so vsi trije izhodiščni dejavniki dobro ocenjeni. Le če so tako možnosti nadzora nad projektom kot podpora timskemu delu ocenjene kot dobre, je lahko izpolnjevanje celotnega kriterija ocenjeno kot dobro;
- *proces.* V kolikor je kateri koli od treh dejavnikov, od katerih je ta kriterij odvisen (t.s. raven avtomatizacije, podpora različnim vlogam v procesu in možnost modeliranja procesov), slabo ocenjen, je izpolnjevanje zahtev procesa ocenjeno kot slabo. Zahteve

procesa so zelo dobro izpolnjenjene, v kolikor so vsi trije izhodiščni dejavniki (ki jih obravnavamo kot približno enako pomembne) dobro ocenjeni;

- *zgradba sistema.* V kolikor sistem slabo izpolnjuje zahteve glede trdoživosti (robustnosti) ali odprtosti ali varnosti, so zahteve glede njegove zgradbe neustrezno izpolnjene. Zgradba sistema je zelo dobro oziroma napredno zasnovana v primeru, ko so vsi trije navedeni osnovni dejavniki, ki imajo podobno težo, ocenjeni kot dobri;
- *poslovna uspešnost.* V primeru ali slabe tržne pozicije ali slabe podpore je poslovna uspešnost slaba. Poslovna uspešnost je zelo visoka, v kolikor sta dejavnika tržna pozicija in podpora dovolj visoko ocenjena. Oba osnovna dejavnika sta približno enako pomembna;
- *tržna pozicija.* V kolikor je podporni sistem (še) neuveljavljen, lahko tržno pozicijo ocenimo kot slabo. V kolikor je podporni sistem uveljavljen ali zelo uveljavljen, oceno kriterija ('Tržna pozicija') lahko znižata ali nižja uveljavljenost proizvajalca ali njegova slaba tehnološko-poslovna perspektiva. Vendar slaba perspektiva proizvajalca sistema sama po sebi še ne pomeni nujno slabe ocene za oceno sestavljenega kriterija, saj v primeru kakovostnega podpornega sistema takšno podjetje v mnogih primerih prevzamejo večja razvojna podjetja, ki nato tak sistem vmestijo v svoj portfelj rešitev. Tržna pozicija je ocenjena kot zelo visoka v primeru, ko je sistem tržno zelo uveljavljen, uveljavljen pa je tudi njegov proizvajalec. Pri tem mora biti tehnološko-poslovna perspektiva proizvajalca ocenjena kot dobra;
- *podpora.* V kolikor stalne izboljšave sistema in tekoče reševanje prijavljenih problemov na sistemu poteka dobro, pri čemer zastopnik nudi tudi kakovostno praktično pomoč pri vpeljavi, je podpora ocenjena kot zelo dobra. V kolikor je en od obeh dejavnikov, od katerih je podpora odvisna, ocenjen kot slab, je celotna podpora ocenjena kot slaba.

4.2.1.2 Značilnosti funkcij koristnosti v zvezi z zahtevami podjetja

V nadaljevanju je podan opis značilnosti funkcij koristnosti za posamezne dejavnike v zvezi s zahtevami podjetja za obravnavani sistem:

- *zahteve podjetja.* V primeru odklonilnega mnenja uporabnikov ali neustrezno izpolnjenih zahtev razvojnega in podpornega sistema podjetja tudi zahteve podjetja niso ustrezno izpolnjene. V primeru omahujočega odnosa uporabnikov bi bilo zaželjeno, da bi bil prehod na nov sistem čim enostavnejši, v nasprotnem primeru bi lahko pri uvajanju naleteli na dodatne težave. Zahteve podjetja bi bile najbolje izpolnjene v primeru, ko bi bile kar se da dobro izpolnjene zahteve podpornega in razvojnega sistema, ob ustreznem zadovoljstvu oziroma podpori uporabnikov in možnostih čim enostavnejšega prehoda na nov sistem;
- *zahteve sistema.* V primeru neustreznih uporabniških vmesnikov, neustrezno izpolnjenih posebnih zahtevah sistema (modularnost, distribuirani razvoj...) ali neustreznih možnostih integracije v celoten podporni in razvojni sistem podjetja, zahteve razvojnega in podpornega informacijskega sistema podjetja niso ustrezno izpolnjene. Vsi trije omenjeni dejavniki imajo pri ocenjevanju možnosti izpolnjevanja zahtev sistema podobno težo. Zahteve sistema so izpolnjene zelo dobro v primeru, ko so vsi trije omenjeni kriteriji ocenjeni z visoko oceno;
- *integracija v sistem.* Integracija v celotni sistem je slaba v primerih, ko ocenjevani podporni sistem ne nudi ustreznih možnosti za povezovanje z razvojnimi in podpornimi sistemi. Tudi v primeru, če podporni sistem zadovoljivo podpira delo le na eni glavni razvojni platformi, to pomeni slabo integracijo v celotni informacijski sistem. Integracija je zelo dobra v primeru, ko podporni sistem ponuja celovito rešitev za obe glavni razvojni

platformi in možnost dela preko Web odjemalcev, poleg tega pa tudi dobre možnosti povezovanja z drugimi obstoječimi razvojnimi in podpornimi sistemi;

- *vmesniki.* V primeru, ko so sistem ponuja neintuitivne in nepregledne vmesnike ali kvečjemu en tip vmesnikov, le-ti ne ustrezajo našim zahtevam. Vmesniki so dobri le v primeru, ko so dobri kar se tiče intuitivnosti, učinkovitost in udobnosti uporabe, pri čemer so vse ključne funkcije sistema podprte z obema glavnima tipoma vmesnikov (interaktivni grafični in ukazni);
- *posebne zahteve.* Pri ocenjevanju izpolnjevanja posebnih zahtev imajo pomembno vlogo vsi trije dejavniki, ki nastopajo v tem sestavljenem kriteriju: tako podpora modularnosti kot podpora upravljanju zahtev za izdelke in podpora geografsko porazdeljenemu razvoju. V kolikor sistem ne ponuja sprejemljive podpore katerega koli od teh dejavnikov, je izpolnjevanje posebnih zahtev neustrezno. V ostalih primerih je izpolnjevanje posebnih zadev zadovoljivo, sprejemljivo. Posebne zahteve so zelo dobro izpolnjene tedaj, ko sta dobro podprta modularnost in geografsko porazdeljen način razvoja, ob sprejemljivi ravni podpore upravljanju zahtev za izdelke;
- *zadovoljstvo.* V kolikor investicija v poslovnem pogledu ne bi bila sprejemljiva ali celovitost rešitve ne bi bila ustrezna, bi bilo stališče večine uporabnikov do uvedbe sistema odklonilno. Zadovoljstvo bi bilo zelo visoko v primeru, ko bi dobili celovito rešitev za področji upravljanja konfiguracij in sprememb proizvodov SI2000, ob sprejemljivi ravni sodelovanja z dobavitelji sistemov. Odnosi z dobavitelji (oziroma svetovalci iz njihovih vrst) imajo razmeroma pomembno vlogo, saj bi bil ob negativnih izkušnjah z njimi odnos uporabnikov do uvedbe sistema v najboljšem primeru omahujoč;
- *ugodnost investicije.* Ocenjujemo, da bi bila investicija najbolj ugodna v primeru, ko njen obseg ne bi bil velik, njena donosnost pa kar se da velika. Pri tem bi bile dobrodošle dodatne finančne ugodnosti s strani dobavitelja sistema. Vse to bi omogočalo kar se da nizko tveganje in minimalni obseg motenj pri poslovanju, ob minimalnih stroških. Če investicija ni donosna, je seveda neugodna, nesprejemljiva, ne glede na njen obseg in morebitne finančne ugodnosti. Višji kot je obseg naložbe, večjo vlogo igrajo (ob pričakovani enaki donosnosti) dodatne finančne ugodnosti;
- *celovitost rešitve.* Rešitev bi bila celovita v primeru, da bi vsaj v obdobju naslednjih petih let ustrezno funkcionalno podprla proces upravljanja konfiguracij in sprememb izdelkov SI2000. V kolikor proces upravljanja konfiguracij in sprememb ne bi bil ustrezno podprt s funkcijami sistema, bi bila celovitost rešitve neustrezna. S celovitostjo rešitve ne bi bili zadovoljni tudi v primeru, ko bi nanjo lahko računali le za obdobje naslednjih dveh let;
- *prehod.* Prehod bi bil netežaven v primeru, ko bi bil sistem enostaven za uporabo in bi bili vanj vgrajeni procesi skladni z našim procesom. Pri tem bi moral biti sistem vsaj delno prilagodljiv, v smislu realizacije novih zahtev. V kolikor bi bil sistem za uporabo kompleksen, bi bil prehod na ta sistem težaven, razen v primeru, ko bi bili vanj implementirani procesi zelo skladni z našim procesom, sam sistem pa zelo prilagodljiv. V kolikor bi bil sistem tog, neprilagodljiv, bi bil prehod netežaven le v primeru njegove enostavne uporabe in skladnosti vanj implementiranih postopkov z našim procesom.

4.2.2 Primerjava pomembnosti kriterijev

Izločilni kriteriji v samem modelu eksplicitno ne nastopajo, vendar pa imajo nekateri kriteriji nekoliko večjo težo od drugih:

- *višina investicije* je pomebna, ker vpliva na višje stroške razvoja in vzdrževanja izdelkov, kar vpliva na ceno in trženje samih izdelkov;
- *celovitost rešitve* je kriterij, povezan s temeljnimi cilji prenove sistema, od tod tudi njegova pomembnost. Celovitost rešitve se nanaša tudi na željo oziroma zahtevo, naj bi sistem v naslednjem nekajletnem obdobju kakovostno služil svojemu namenu. To vpliva tudi na zaupanje in zadovoljstvo tako uporabnikov sistema kot vodstva;
- *težavnost prehoda* se nanaša na to, kako težaven bi bil prehod. V primeru velikih težav lahko pričakujemo nižjo kakovost realizacije rešitve, zamike pri izvedbi projekta uvajanja sistema in nižjo raven zadovoljstva različnih ravni uporabnikov;
- *funkcionalnost* je kriterij, ki ponazarja splošno tehnično uporabnost podpornega sistema;
- *zgradba sistema* je pomembna za učinkovito podporo upravljanju življenjskega cikla proizvodov v geografsko porazdeljenih sistemih, za varno delo s podatki, za zagotavljanje celovitosti in skladnosti podatkov ter za zagotavljanje možnosti povezovanja podpornega sistema z drugimi sistemi, tako znotraj kot izven podjetja;
- *integracija v sistem* je pomembna, ker ima podjetje vizijo združevanja različnih informacijskih podsistemov v celovit informacijski sistem o proizvodih, kar bi omogočalo še uspešnejše in učinkovitejše upravljanje z izdelki;
- *zadovoljstvo uporabnikov.* Podporni sistem je lahko učinkovit in lahko v celoti zaživi le, če so uporabniki z njim zadovoljni, vodstvo pa prepričano, da lahko z uporabo takšnega sistema zagotovi doseganje ustreznih poslovnih učinkov. Nezadovoljstvo uporabnikov lahko pomeni povečane stroške pri uvajanju orodij v proces (pasivnost uporabnikov), posredno pa tudi manjšo učinkovitost orodij pri podpori procesa;
- *skladnost procesa,* ki ga podpira sistem, z zahtevami za razvojni proces in proces upravljanja konfiguracij izdelkov in njihove PO v podjetju, zelo vpliva na to, v kolikšni meri bomo lahko izkoristili potenciale podpornega sistema;
- dejavnik *prilagodljivost* ponazarja možnost prilagajanja sistema. Ta kriterij bo pomemben predvsem v fazi vzdrževanja podpornega sistema, saj se v zahteve uporabnikov in samega procesa neprenehoma spreminjajo;
- *perspektiva.* Tehnološka in poslovna perspektiva proizvajalca sistema je pomemben kriterij. Odraža se npr. v tem, v kolikšnem podjetje, ki proizvaja dani sistem, razvija nove, perspektivne tehnološke koncepte na področju upravljanja konfiguracij in sprememb proizvodov. V kolikor proizvajalec sistemov nima perpektive, se veča verjetnost, da bo sistem v kratkem izginil s tržišča ali pa bo tako ali drugače nadomeščen z drugim sistemom.

5. Uporaba odločitvenega modela

V tem poglavju so predstavljene in ocenjene nastopajoče variante za sistem za upravljanje konfiguracij izdelkov in njihove programske opreme. Na tržišču trenutno ni mogoče dobiti podpornega sistema "v enem kosu", ki bi ustrezal vsem ključnim zahtevam vodenja konfiguracij in sprememb izdelkov z vloženimi računalniškimi sistemi, kakršne izdeluje podjetje Iskratel. Kljub temu, da obstaja trend povezovanja funkcionalnosti PDM in SCM sistemov, do zdaj še ni prišlo do zadovoljive združitve v povezan in celovit sistem, ki bi ustrezal visokim zahtevam razvojnih sistemov s področja telekomunikacij.

Zato je potrebno vzeti osnovne SCM in PDM sisteme in jih povezati med seboj in z drugimi obstoječimi sistemi za podporo poslovanju v celovit sistem, ki bo kar najbolje ustrezal zahtevam

poslovnega procesa v podjetju in kar se da učinkovito podpiral upravljanje življenjskega cikla proizvodov. V skladu z navedeno ugotovitvijo so v nadaljevanju poglavja navedeni:

- način, kako smo izbrali aktualne variante;
- nato so predstavljeni elementarni SCM in PDM sistemi, ki sestavljajo variante;
- sledi kratka predstavitev variant oziroma izbranih sestavljenih sistemov;
- v zadnjem podpoglavju so predstavljeni rezultati vrednotenja variant.

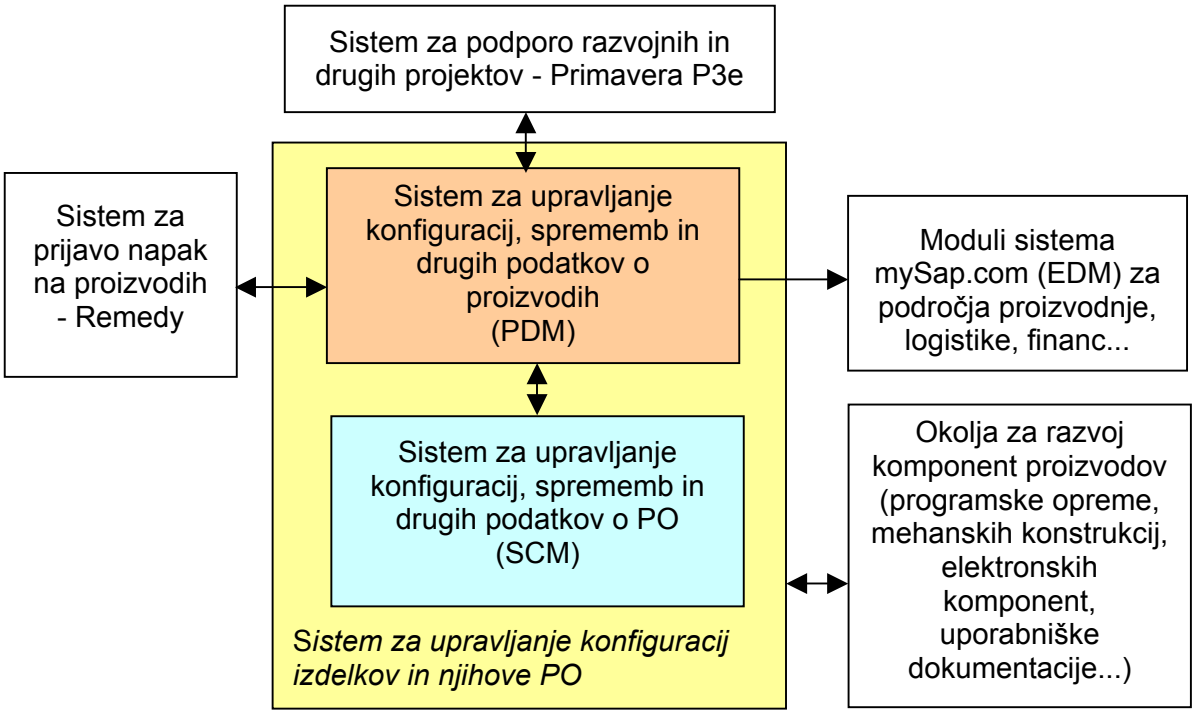
5.1 Izbira variant

Pri izbiri variant smo upoštevali naslednje kriterije, med katerimi so nekateri navedeni v prejšnjih poglavjih, drugi v interni dokumentaciji podjetja (Frey-Pučko, 2002, str. 15) itd.:

- upoštevali smo že omenjene zahteve po izboljšavah v procesu in orodjih;
- upoštevali smo dejstvo, da trenutno na tržišču ni enega samega podpornega sistema oziroma modula, ki bi bil primeren tako za upravljanje konfiguracij in sprememb celotnih proizvodov, sestavljenih iz različnih tehnologij, kot tudi za upravljanje konfiguracij in sprememb programske opreme;
- predpostavljali smo, da bo PDM modul sistema nase prevzel podporo celovitega, povezanega upravljanja konfiguracij in sprememb izdelkov in njihovih komponent (vključno z vrhnjimi kosovnicami PO), medtem ko naj bi SCM modul podprl upravljanje konfiguracij PO, vključno z nadzorom verzij programskih gradnikov in paketov;
- upoštevano je bilo dejstvo, da je bilo podjetje Sherpa PIMS prevzeto s strani podjetja SDRC in to pozneje s strani EDS, ki PDM funkcionalnosti ponuja v okviru komponent svojega sistema Teamcenter. Podjetje EDS ponuja vsem Sherpinim uporabnikom finančne in druge ugodnosti ob prehodu na sistem Teamcenter, ne zagotavlja pa več vzdrževanja za Sherpin sistem. Predpostavili smo, da bo Sherpa PIMS, ki je temelj našega PDM sistema SOPRAN, ob trenutnem načinu vzdrževanja s strani interne skupine sistemskih inženirjev uporaben še največ dve leti;
- v podjetju se na nekaterih področjih poslovanja (finance, nabava...) uporablja sistem mySAP.com, ki ima tudi poseben modul za upravljanje življenjskega cikla proizvodov;
- pri komercialnih SCM sistemih smo se usmerili na sistema, ki sta se v prejšnjih ocenjevanjih pokazala kot tista, ki med tovrstnimi sistemi najbolje ustrezata našim zahtevam na področju upravljanja konfiguracij PO. To sta ClearCase in CM Synergy;
- pri najpomembnejših poslovnih partnerjih se za področje upravljanja konfiguracij PO najpogosteje uporablja SCM sistem ClearCase. Upoštevano je stališče, da je ta sistem za podjetje zanimiv le v kombinaciji s svojim procesnim modelom UCM;
- v ustrezni meri so bila upoštevana stališča in mnenja tako vodstva poslovne enote kot uporabnikov podpornih sistemov;
- upoštevana je bila tudi naprednost konceptov in rešitev, vgrajenih v sisteme.

Pri izbiri izbiri in obravnavi variant je upoštevana tudi vmeščenost sistema za upravljanje konfiguracij izdelkov in njihove PO v celotno okolje oziroma sistem za podporo poslovanja z življenjskim ciklom izdelkov SI2000 (glej sliko 19).

Slika 19: Vmeščenost sistema za upravljanje konfiguracij izdelkov in njihove PO v okolje za podporo poslovanja z življenjskim ciklom izdelkov SI2000



Ob upoštevanju navedenih kriterijev smo prišli do naslednjega nabora variant:

- nadaljevanje razvoja internega sistema, temelječega na obstoječem internem SCM sistemu in PDM Sopran-u;
- povezava sistemov Teamcenter in ClearCase;
- povezava sistemov mySAP.com (oziroma modula mySAP PLM) in CM Synergy. To je komplementarna možnost glede na prejšnjo (Teamcenter in ClearCase);
- povezava sistemov mySAP.com (oziroma modula mySAP PLM) in ClearCase;
- povezava sistemov Teamcenter in nadaljevanja razvoja obstoječega internega SCM.

V nadaljevanju so najprej predstavljeni osnovni, elementarni CM sistemi. Nato so predstavljene še same (sestavljene) variante.

5.2 Sistemi za podporo upravljanja konfiguracij izdelkov in njihove programske opreme

V nadaljevanju so opisani sistemi za podporo upravljanju konfiguracij izdelkov ali njihove programske opreme, ki sestavljajo zgoraj navedene variante. To so:

- sistem SCCS;
- Sherpa PIMS;
- Rational ClearCase;
- EDS Teamcenter;
- mySAP PLM;
- Telelogic CM Synergy.

SCCS, ClearCase in CM Synergy so v osnovi namenjeni nadzoru verzij in upravljanju konfiguracij programske opreme. Pri tem SCCS zadovoljivo podpira le nadzor verzij, ClearCase in CM Synergy pa sta vrhunska komercialna SCM sistema.

Ostali trije sistemi so namenjeni upravljanju življenjskega cikla proizvodov. Sherpa PIMS je "mrtva" aplikacija, brez zagotovljene nadaljnje podpore, medtem ko EDS Teamcenter in mySAP PLM spadata med najkakovostnejše sisteme s PDM/PLM funkcionalnostjo.

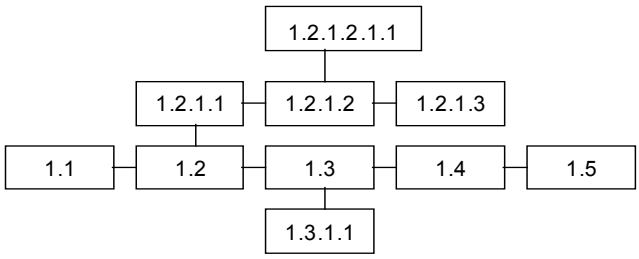
5.2.1 Orodje SCCS

SCCS je množica orodij v sklopu operacijskega sistema HP-UX. Namenjen je nadzoru nad verzijami programskih gradnikov, ki vsebujejo standardne ASCII znake (npr. datotek z izvorno kodo programov). SCCS pokriva le nekaj funkcionalnosti v zvezi z upravljanjem konfiguracij PO in je značilen predstavnik modela odjava/prijava. V kolikor uporabnik želi doseči zadovoljivo raven podpore, mora izvesti nadgradnjo za nekatere ključne funkcije upravljanja konfiguracij PO, npr. nadzor nad strukturo programskih paketov. Kako je bila nadgradnja izvedena v Iskratelu, je orisano v razdelku 3.1.2.3.

5.2.1.1 Osnovne funkcije in mehanizmi orodja SCCS

To orodje poleg *nadzora verzij* datotek podpira tudi *vejitve* (angl. branch) datotek. Pomembnejše funkcionalnosti sistema SCCS: avtomatsko sledenje sprememb, narejenih nad tekstovno datoteko; dostop do zadnje oziroma poljubne verzije datoteke; avtomatski nadzor nad tem, kdo je datoteko popravljal; shranjevanje nekaterih pomembnejših metapodatkov v zvezi s spreminjanjem programskih gradnikov, kot npr.: datum in lokacija spremembe; avtor spremembe; kratek opis spremembe.

Slika 20: Primer strukture in označevanja verzij programskega gradnika v SCCS



Vir: Interna dokumentacija podjetja

Ko se neka datoteka shrani v SCCS, se vse njene verzije ter nekatere dodatne informacije (metapodatki) shranijo v tako imenovani s-datoteki (vse SCCS datoteke imajo predpono “s.”). To je datoteka, v kateri so shranjene vse verzije dane tekstovne datoteke. V njej se dejansko shranjujejo le razlike (delte) med posameznimi verzijami. Celotna vsebina se shrani le za prvo verzijo. Tak način shranjevanja običajno prihrani prostor v primeru izvornih programskih datotek, saj se pri spremembah navadno spremeni le razmeroma majhen del njihove vsebine.
Za vsako verzijo gradnika obstaja enoličen *SCCS identifikator* (krat. SID), sestavljen iz *izdaje* (angl. release) in *stopnje* (angl. level) - primer: 1.3. Ponavadi se izdaja ohranja, stopnja pa se ob vsaki novi verziji poveča. Izdaja naj bi se spremenila le, če pride v gradniku do večje spremembe. SID za prvo verzijo je 1.1, nato pa se avtomatsko povečuje.
Nad SCCS datotekami je mogoče izvajati naslednje osnovne operacije:

- *Odjava verzije iz SCCS brez rezervacije.* To storimo, kadar gradnika ne nameravamo spreminjati, temveč želimo le npr. pogledati njegovo vsebino ali ga prevesti. Operacijo izvedemo nad ustrezno SCCS datoteko. Kot rezultat operacije dobimo zahtevano verzijo gradnika v delovni mapi.

- *Odjava verzije iz SCCS datoteke z rezervacijo.* To storimo, ko želimo datoteko spremeniti in jo nato vgraditi nazaj v SCCS datoteko. Rezultat te operacije je zahtevana verzija gradnika v tekoči datotečni mapi. Poleg tega se nad to verzijo gradnika izvede rezervacija; to pomeni, da ostali uporabniki te verzije gradnika ne morejo spreminjati. Istočasno ima lahko določeno verzijo gradnika rezervirano le en uporabnik.
- *Sprostitev verzije gradnika.* Rezultat te operacije je sprostitev že izvedene rezervacije določene verzije programskega gradnika (glej prejšnjo operacijo). Izvedba te operacije pride npr. v poštev, kadar se uporabnik med spreminjanjem gradnika premisli in se odloči, da sprememb ne bo vključil v knjižnico. S sprostitivijo verzije le-ta postane ponovno dosegljiva vsem uporabnikom. Velja omejitev: operacija sproščanja je možna le v paru z operacijo odjave datoteke iz SCCS z rezervacijo.
- *Prijava nove verzije gradnika nazaj v knjižnico.* Rezultat te operacije se kaže v spremenjeni, dopolnjeni SCCS datoteki, poleg tega pa tudi v sprostitvi rezervacije izhodiščne verzije programskega gradnika. Spremenjeni datoteki iz tekoče datotečne mape se priredi nova verzija gradnika v SCCS datoteki. Vsebina nove verzije gradnika se skupaj z metapodatki o spremembi shrani v SCCS datoteko. Velja omejitev kot pri prejšnji operaciji: operacija vključitve v knjižnico je možna le v paru z operacijo odjave datoteke iz SCCS z rezervacijo.

5.2.2 Sherpa PIMS

Aplikacija Sherpa PIMS je izdelek podjetja Sherpa Corporation. Ta podporni sistem se na tržišču ne prodaja več, saj je bilo podjetje Sherpa prevzeto s strani SDRC, ponudnika orodja Metaphase. Tega je pred kratkim prevzelo podjetje EDS, ki funkcionalnosti Metaphase-a vključuje v module svojega sistema Teamcenter.

5.2.2.1 Značilnosti zgradbe sistema

Aplikacija Sherpa PIMS ima zgradbo tipa strežnik/odjemalec. Medtem ko strežnik teče na sistemu HP-UX, so odjemalci lahko tako na sistemih HP-UX kot Windows.
Osnovo sistema predstavlja odlagališče podatkov in metapodatkov. Ta temelji na Oraclovi relacijski podatkovni bazi in datotečnem sistemu. V slednjem se shranjuje vsebina gradnikov. Podatkovna baza je namenjena vodenju in nadzoru nad metapodatki. Vsebuje vnaprej določene tabele za osnovne tipe zapisov, povezave med njimi, vodenje podatkov o življenjskem ciklu zapisov posameznih tipov in drugih atributov entitet proizvodov in procesa.
Nad omenjenim odlagališčem podatkov in metapodatkov je postavljena t.im. aplikacija PIMS, v kateri so implementirane funkcije oziroma operacije, ki jih sistem podpira. Pred in po uvedbi sistema v obratovanje so bile izvedene različne prilagoditve PDM funkcij in dodane razne nove operacije (npr. izdelava poročil različnih tipov). Prilagoditve so v glavnem izkoriščale mehanizem t.im. alertov (alarmov, opozorilnikov). Dodane operacije se tako izvedejo pred, ob ali po dogodkih danega tipa. Uporabljen je jezik DMS, ki je namenjen izvajanju prilagoditev, tako za aplikacijske kot za uporabniške vmesnike.

5.2.2.2 Predstavitev osnovnih funkcij sistema

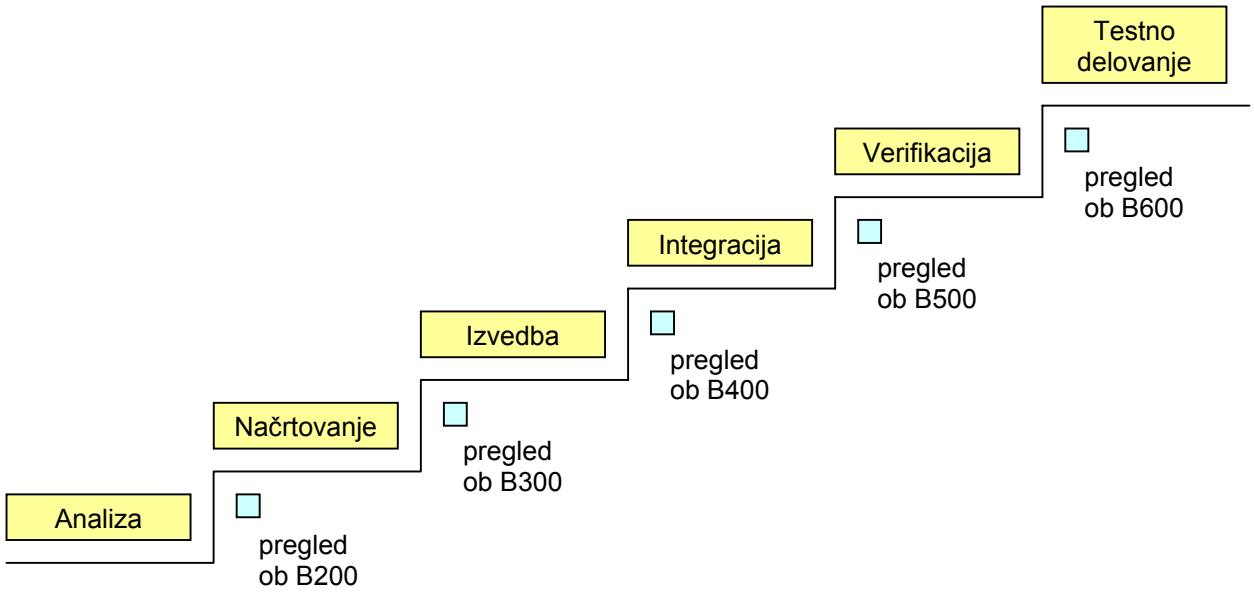
Sherpa PIMS je v osnovi namenjena upravljanju proizvodov skozi njihov življenjski cikel in vsebuje naslednje pomembnejše funkcije:

- nadzor verzij/revizij različnih tipov gradnikov in kosovnic;
- upravljanje konfiguracij izdelka;
- upravljanje inženirskih sprememb nad izdelki in

- upravljanje dokumentacije.

Sherpa PIMS pozna naslednje tipe zapisov: PRODUCT, PART, ECR (krat. za Engineering Change Request), ECO (krat. za Engineering Change Order), ECN (krat. za Engineering Change Notice), DOCUMENT. Za te zapise se vodijo osnovni, v naprej določeni atributi, nanje je možno pripenjati datoteke, lahko vsebujejo kazalce na druge zapise, prav tako pa se zanje lahko vodijo življenjski cikli različnih tipov, katerih sestavni del so običajno postopki rezervacije, izdajanja, obravnave in potrjevanja (glej sliko 21).

Slika 21: Življenjski cikel za zapis tipa PRODUCT



Vir: Interna dokumentacija podjetja

Med temi tipi zapisov uporabnik lahko vzpostavlja povezave naslednjih tipov: "described by", "provides", "raised against", "closes", "issues", "uses", "releases". Struktura zapisov oziroma možnosti povezovanja različnih tipov med seboj so prikazani na sliki 17.

Sistem s svojimi PDM funkcijami podpira tako vodenje strukture proizvodov (uporabe kosovnic in sestavnic) kot strukture verzij in revizij na nivoju posameznega gradnika. Za povezave je možno voditi atribute, ki omogočajo npr. lažje razvrščanje zapisov oziroma entitet in njihovo povezovanje na ustrezne funkcionalnosti proizvodov. Omenjene operacije ponujajo dobro osnovo za obvladovanje strukture izdelkov.

Sherpa PIMS zagotavlja tudi osnovno podporo upravljanju konfiguracij programske opreme (SCM). Razvijalcu omogoča delo v njegovem privatnem oziroma testnem delovnem okolju, pri čemer lahko uporablja vse funkcije podatkovnega odlagališča. Za programske gradnike je, tako kot za vse druge tipe zapisov, mogoče podpreti njihov življenjski cikel. Pri tem lahko uporabnik neposredno izkorišča vse podatke, ki jih sistem vodi za dani izdelek. Brez dodatnih vložkov je mogoče izvesti tudi gradnjo datotečnega okolja, predstavljenega s kosovnicami, ki ponazarjajo datotečno strukturo programske opreme. Pri delu na nivoju posameznih gradnikov proizvodov (vključno z dokumentacijo) so podprte naslednje funkcije: arhiviranje podatkov in metapodatkov, z uporabo mehanizma odjava/prijava; vodenje izdaj gradnikov; obravnava in potrjevanje gradnikov; doseganje vsebine prek intraneta; povezave na druge gradnike, kosovnice ali produkte.

Ena najmočnejših funkcij Sherpinega sistema je podpora upravljanja inženirskih sprememb na izdelkih. Kako Sherpa PIMS podpira ta proces, je podrobneje opisano v razdelku 3.1.3.

5.2.3 Rational ClearCase

Orodje ClearCase razvija podjetje Rational Software. Razvojni laboratoriji za ClearCase so v Lexingtonu (ZDA). Podjetje Rational razvija ne le sistemov orodij za podporo celotnega življenjskega cikla (predvsem programskih) izdelkov, temveč celovite procesne pristope na področju razvoja, uvajanja in vzdrževanja teh izdelkov. Njihov procesni pristop RUP (krat. za Rational Unified Process) je predstavljen v prilogi E.

Na področju upravljanja konfiguracij in sprememb programske opreme razvijajo procesni pristop UCM (krat. za Unified Change Management). Obstajata dve varianti orodja ClearCase - brez UCM in z vgrajenim UCM. Ker je razumevanje UCM za delovanje ClearCase-a pomembno, je v nadaljevanju opis UCM podan pred opisom sistema ClearCase.

Glavni Rationalovi moduli za upravljanje konfiguracij in sprememb so:

- Rational ClearCase, namenjen upravljanju konfiguracij programske opreme;
- Rational ClearQuest, namenjen upravljanju nalog, zahtev za spremembo in sledenje problemov (ang. defect tracking);
- MultiSite za upravljanju paralelnega razvoja skupin na različnih geografskih lokacijah.

5.2.3.1 Rational UCM

Rational UCM (Rational Unified Change Management) je koncept enotnega upravljanja sprememb na temelju aktivnosti. Podprt je z orodjema ClearCase in ClearQuest (glej prilogo F).

Slika 22: Filozofija pristopa UCM - povezanost projektnih aktivnosti z delovnimi proizvodi

Vir: White, 2000, str. 52

Osnovni objekti in mehanizmi, ki jih izkorišča UCM (Marand, 2001, str. 12):

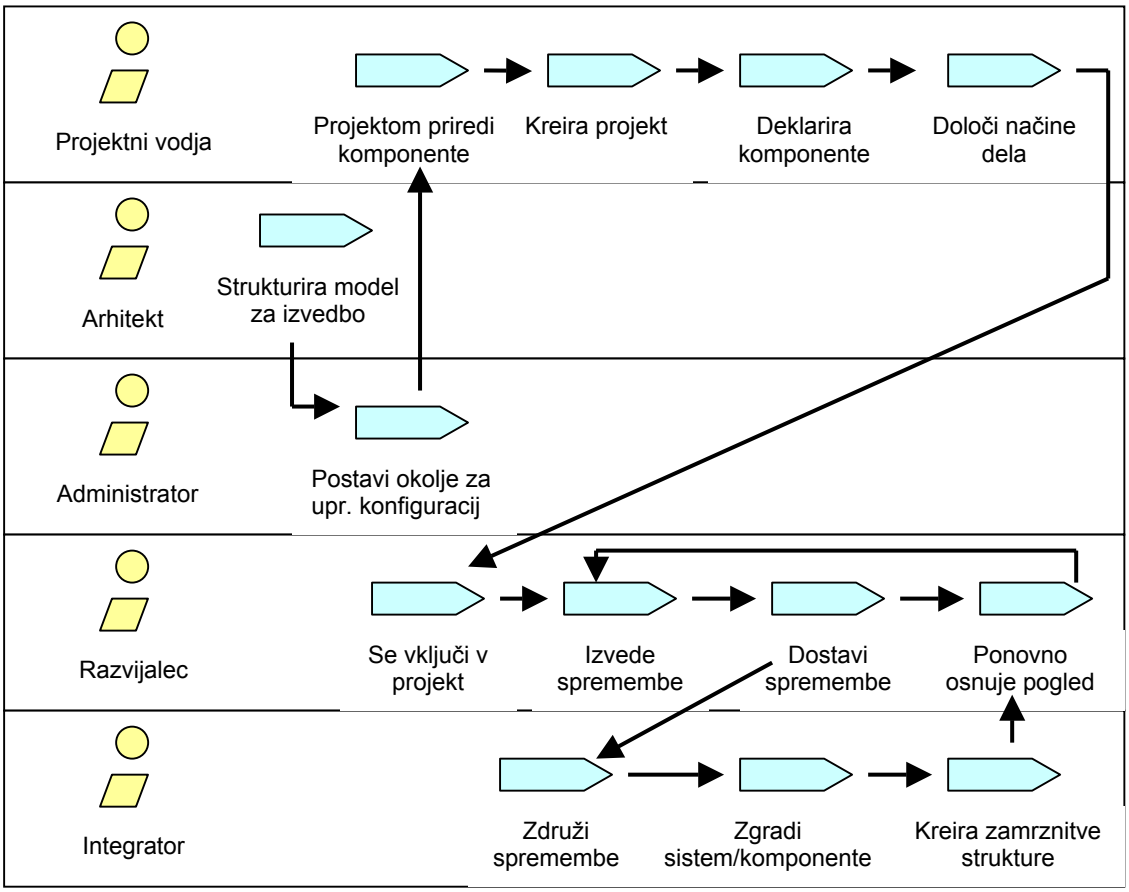
- *projekt* - objekt, ki vsebuje informacijo o konfiguraciji, potrebno za nadzorovano izvedbo določene razvojne operacije (npr. izdajanja izdelka). Projekt vsebuje eno skupno delovno področje in množico privatnih delovnih okolij;
- *aktivnost* - beleži množico datotek, ki jih uporabnik ustvari ali spremeni z namenom izvedbe razvojne naloge;
- *komponenta* - mehanizem za poenostavitev vodenja organizacije datotek in datotečnih map. Elementi (datoteke in mape), ki jih uporabnik združi v komponento, običajno predstavljajo pouporabljiv kos zgradbe sistema (programskega izdelka);
- *izhodiščna konfiguracija (angl. baseline)* - določa eno verzijo vsakega elementa v komponenti. Izhodiščne konfiguracije se shranjujejo posebej za vsak projekt;
- *množica gradnikov v zvezi s spremembo (angl. change set)* - množica gradnikov, povezana z neko aktivnostjo;

- *tok (angl. stream)* - objekt, ki vzdržuje seznam aktivnosti in določa, katere verzije elementov naj se pojavijo v uporabnikovem delovnem okolju - *pogledu* (angl. view).

UCM projekt poleg zgoraj omenjenih vsebuje tudi informacije, povezane z integracijskim tokom projekta (t.j. tokom, namenjenemu izvajanju integracije programske opreme v danem projektu), načini dela v projektu (npr. način izdajanja sprememb s strani uporabnikov) in seznami komponent, dosegljivih za ta projekt. Vsak od uporabnikov ima svoj razvojni tok. Ta je sestavljen iz primernih izhodiščnih konfiguracij za vsako od komponent in iz delovnega okolja (angl. view) uporabnika. Za projekt obstaja le en integracijski tok.

5.2.3.1.2 Delo s ClearCase UCM - uporabniški model

Slika 23: Potek dela petih vlog, vključenih v projekt UCM; ključne vloge so razvijalec, integrator in vodja projekta



Vir: Marand, 2001, str. 14

ClearCase je zasnovan tako, da zagotavlja lahek način vpeljave in uporabe prilagodljivega procesa. UCM določa vsaj tri ključne uporabniške vloge za projekt:

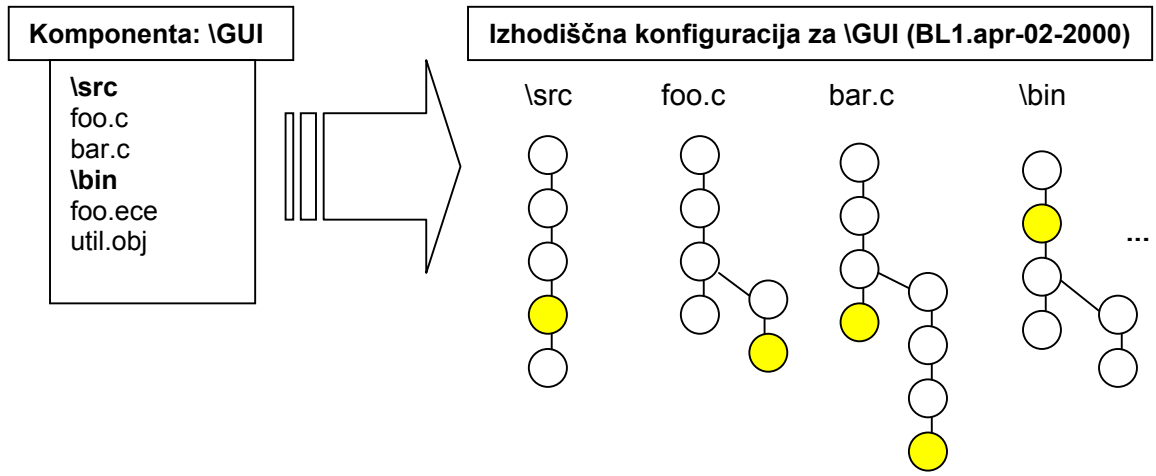
1. *razvijalec* - odgovoren je za razvoj gradnikov sistema (npr.: HTML dokumentov, izvornih gradnikov v C++, Javi...). Pretok dela za razvijalca (glej sliko 23) vsebuje: vključitev v obstoječi projekt, izvedba sprememb v zvezi z aktivnostjo (npr. odprava napake), uskladitev delovnega toka s tekočo priporočeno izhodiščno konfiguracijo in dostava izvedenih sprememb v integracijski tok;
2. *izvajalec integracije* - zagotavlja povezovanje dejavnosti in rezultatov, ki se izvajajo ali nastanejo pri vzporednem izvajanju aktivnosti, v integracijski tok. Po izvedbi testiranja se na podlagi tega naredi ustrezna izhodiščna konfiguracija dane komponente. Ta vloga zahteva dobro razumevanje projekta in poznavanje razdelitve dela;

3. *vodja projekta* - ustvari namen in cilje projekta, vodi tekoče projektne aktivnosti, poverja aktivnosti, določa komponente, ki bodo del projekta, in določa načine dela v projektu.

5.2.3.1.3 Model napredovanja izhodiščne konfiguracije

UCM uporablja izhodiščne konfiguracije za komponente. Ta izhodiščna konfiguracija je množica verzij datotek, ki se ustvari in vzdržuje na ravni komponente. UCM v zvezi s tem zagotavlja možnost označitve 'koristnosti' za te izhodiščne konfiguracije, pri čemer ji priredi neko raven, ki nekaj pomeni vodju projekta, izvajalcu integracije ali razvijalcu. Sam po sebi ClearCase vsebuje pet nivojev napredovanja - osnovna zamisel je v tem, da izhodiščna konfiguracija začne svoj življenjski cikel na spodnjem nivoju in nato napreduje na višje nivoje po izvedbi določenih kontrolnih operacij. V naprej nastavljeni nivoji so: "zavrnjen", "začeten", "zgrajen", "pretestiran", "izdan". Kot dodatek h konceptu napredovanja izhodiščnih konfiguracij obstaja zamisel o uporabi t.im *priporočeni izhodiščni konfiguraciji.* Izhodiščna konfiguracija postane za razvijalca priporočena v tem smislu, da jo ta lahko uporabi kot osnovo, izhodišče za nadaljne spremembe. Pogoj za to je, da taka konfiguracija doseže nek nivo potrditve. Primer: kakršna koli izhodiščna konfiguracija postane priporočena, ko pride vsaj na nivo "testiran". V smislu uporabe tega pristopa je zelo priporočljivo, da je nivo "testiran" najnižji zahtevani nivo za izhodiščno konfiguracijo. S tem se ta konfiguracija lahko uporabi kot izhodišče za nadaljnje spremembe. Izhodiščna konfiguracija mora biti dobro pretestirana, da služi svojemu namenu in da se lahko uporabi v vseh uporabniških tokovih, vse dokler ni na voljo naslednja (priporočena) izhodiščna konfiguracija, potrjena na ustreznem nivoju.

Slika 24:Komponenta je konfiguracija, sestavljena iz verzij gradnikov



Vir: Marand, 2001, str. 15

Vsaka izhodiščna konfiguracija ima enolično ime, ki običajno vsebuje časovno oznako. Čeprav je ta časovna oznaka vodena že med atributi izhodiščne konfiguracije, jo je zaradi boljše preglednosti priporočljivo vključiti v ime konfiguracije. Primerna oblika imena za izhodiščno konfiguracijo bi tako bila: *BL<ime ali številka gradnje>.<časovna oznaka>* (npr. BL8640.02-apr-2000.14:45:03).

5.2.3.1.4 Uporaba komponente v večih projektih

Ker so komponente v UCM neodvisne od projektov in v projekte vključene le skozi projektni VOB, lahko dano komponento z lahkoto vključujemo in uporabljamo v različnih projektih. Bolj natančno: če je potrebno, je komponenta lahko namenjena uporabi (stanje: "read") v enem projektu in spreminjanju v drugem projektu (stanje: "read"/"write"). To je odvisno npr. od tega, ali jo v danem projektu le uporabljamo ali pa jo v tem projektu tudi načrtujemo in razvijamo.

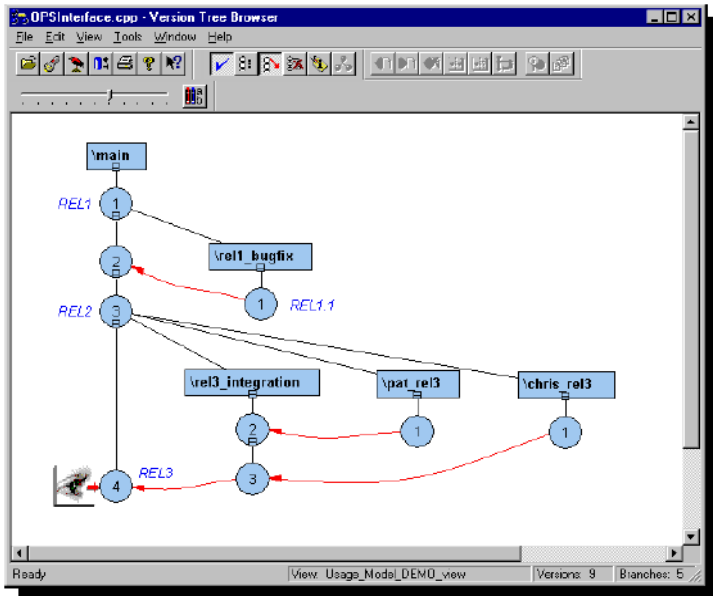
Izhodiščne konfiguracije komponente so shranjene v projektnih VOB-ih, kar omogoča, da isto komponento s stanjem "read"/"write" uporabljamo v različnih VOB-ih, vendar pa se v različnih VOB-ih zanjo izdelujejo tudi različne izhodiščne konfiguracije. S pomočjo orodja ClearCase Project Explorer imamo možnost dodajati komponente v UCM projekt ali jih brisati iz njega.
V kolikor se uporablja orodje MultiSite, razvojna skupina lahko dela na neki oddaljeni lokaciji, integracijski tok projekta pa se upravlja na drugih strežnikih (oziroma kopijah VOB-ov), ne na izvornih razvojnih tokovih. V teh primerih omenjeni razvijalci ne morejo izvesti operacije dostavljanja (angl. deliver) spremenjenih delov izdelka v integracijski tok, temveč mora to storiti projektni vodja. UCM predvideva različico operacije dostavljanja, t.im. *oddaljeno dostavljanje*, ki začne izvajati operacijo dostavljanja, vendar pri tem ne izvaja operacije združevanja (angl. merge) verzij. To mora storiti izvajalec v vlogi vodja projekta.

5.2.3.2 Zgradba in funkcionalnosti orodja Rational ClearCase

5.2.3.2.1 Značilnosti zgradbe in ključni koncepti v zvezi z uporabo

Opis v nadaljevanju temelji pretežno na Rational ClearCase verzija 4 ali poznejših verzijah. ClearCase je sistem z arhitekturo strežnik/odjemalec in je objektno zasnovan. Uporabnik dela s tipi SCM objektov (t.j. gradnikov, sprožilcev, oznak...). Za dane tipe nato lahko kreira posamezne primerke (instance). Tako tipom kot primerkom se lahko določajo in spreminjajo ustrezni atributi, ko pa jih ne potrebujemo več, se (v večini primerov; izjeme so nekateri vnaprej določeni tipi) lahko zbrišejo.
Strežniki običajno tečejo na različicah operacijskih sistemov Unix in Windows, odjemalci pa na obeh navedenih platformah, poleg tega pa tudi v Web okolju. Inštalacija ClearCase odjemalcev mora biti izvedena eksplicitno, za vsakega odjemalca posebej. ClearCase-ove licence so namreč vezane na odjemalce.
Da bi razumeli zmožnosti orodja ClearCase, je potrebno razumeti nekaj ključnih konceptov, na katere se opira to orodje, npr. (Marand, 2001, str. 6): delitev in izolacija razvojnih nalog, ustvarjanje vejitev in združevanje verzij gradnikov, podpora gradnji in izdajanju programske opreme ipd. Odlagališče, kjer so elementi shranjeni, je VOB (kratica za angl. Versioned Object Base). Pogled (angl. view) predstavlja delovno okolje, ki ga ClearCase upravlja za uporabnika, pri čemer nadzoruje delovni kontekst in hkrati zagotavlja okolje za spreminjanje elementov in gradnjo PO. Nadzorovani elementi konfiguracije (datoteke in mape) imajo sami zase nekaj zanimivih lastnosti. Med njimi je t.im. drevo verzij (glej sliko 25). Drevo verzij je urejena zbirka verzij danega elementa. Verzije so urejene zaporedno po logičnih strukturah, ki jim pravimo vejitve (angl. branches).
ClearCase ima vgrajen mehanizem za uveljavljanje različnih načinov dela za različne razvojne vloge, za sledenje napak, pretok podatkov in dela, modele in cikle napredovanja ter stanja procesa (Marand, 2001, str. 6). Paralelni razvoj gradnikov je v ClearCase-u podprt z uporabo: različnih delovnih okolij oziroma pogledov (angl. view) za različne uporabnike; možnostjo vejitev elementov (tako datotek kot map, direktorijev); z rezerviranimi ali nerezerviranimi odjavami (angl. check-out); z operacijo zlivanja dveh verzij tekstovnih gradnikov (angl. merge).

Slika 25: Drevo verzij



Vir: Marand, 2001, str. 7

5.2.3.2.2 Skladišče podatkov (VOB) in datotečni sistem MVFS

VOB (krat. za Versioned Object Base) je namenjen skladiščenju podatkov, ki nastopajo v danem projektu. Smiselno je npr., da se kreira po en VOB za en projekt ali npr. en VOB za eno projektno lokacijo. Uporabniki pri svojem delu običajno uporabljajo več VOB-ov hkrati. Vsak VOB vsebuje ne le vsebine gradnikov oziroma njihovih verzij, temveč tudi metapodatke, ki so shranjeni v VOB-ovi podatkovni bazi. Vrsti VOB-ov:

- *javni VOB-i:* to je običajna vrsta VOB-ov, ki se uporabljajo pri delu v razvojnem projektu. Shranjeni so na ustreznih ClearCase-ovih strežnikih;
- *privatni VOB-i:* uporabnik jih kreira na lastnem sistemu in se lahko npr. uporabljajo za testne namene.

MVFS (krat. za MultiVersioned File System) je poseben datotečni sistem, prilagojen za delo z VOB-i in pogledi. Večino podatkov, ki jih uporabnik v svojem okolju potrebuje za delo z VOB-i, je dobljena preko MVFS. MVFS mora biti nameščen na vsakem ClearCase-ovem odjemalcu. Na strežniku ga ni potrebno instalirati. Zažene se ob zagonu sistema. Najbolj značilna za ta datotečni sistem je njegova dinamičnost. Uporabniku preko uporabe pogledov in njihovih lastnosti med drugim omogoča transparentni način dela na izvorni strukturi paketa.

5.2.3.2.3 Delovna okolja in njihove značilnosti

ClearCase-ov pogled (angl. view) predstavlja delovno okolje uporabnika. V pogledih so fizično shranjene: odjavljene datoteke, ki jih uporabnik trenutno spreminja; uporabnikove privatne datoteke in izpeljani objekti. Pogledi so nekakšne "leče", skozi katere uporabnik gleda na podatke projekta. Bistveni del vsakega pogleda je njegov "config spec" (okrajšava za 'specifikacijo konfiguracije'). Z njim uporabnik določi, katere verzije gradnikov želi imeti vključene v svoj pogled. V svoje delovno okolje lahko npr. vključi ustrezne verzije vseh tistih gradnikov, ki jih namerava spremeniti.

Uporabniki običajno delajo v *dinamičnih pogledih*, ki podpirajo že omenjeni transparentni način dela na strukturi programskega paketa. Uporabnik lahko ustvari in uporablja po več pogledov - vsak od njih npr. namenjen svoji nalogi. V razmerah, ko je računalniška mreža dalj časa počasna, lahko postane delo z dinamičnimi pogledi precej zamudno. Ob takšnih priložnostih ima uporabnik možnost kreiranja in uporabe t.im. *posnetkovnih pogledov* (angl. snapshot view).

V nijih dostop do gradnikov ne poteka dinamično preko lokalne računalniške mreže kot pri dinamičnih pogledih, temveč se pri ažuriranju posnetkovnega pogleda (v skladu z danim "config spec"-om) na uporabnikovem računalniku izdela fizična kopija izhodiščne konfiguracije. Za to je potreben določen čas. Vendar se pozneje operacije v posnetkovnih pogledih izvajajo precej hitreje kot v dinamičnih pogledih - večina akcij se namreč izvaja na lokalnem sistemu. To tudi pomeni, da vpogled v spremembe ostalih uporabnikov v splošnem ni več ažuren. V kolikor uporabnik želi imeti tekoč pregled nad stanjem programskega paketa, mora sam poskrbeti za dovolj pogosto osveževanje vsebine svojega posnetkovnega pogleda.

5.2.3.2.4 Gradnja programske opreme

Za gradnjo paketov na Windows NT se uporablja orodje omake (na Unixu: ClearMake z zelo podobnimi lastnostmi). To orodje je združljivo s standardnim Unixovim orodjem make, pri čemer vsebuje dodaten nabor pravil in makrojev. Pri gradnji z orodjem omake nastanejo izpeljani objekti, ki so lahko tako končni, ciljni objekti (angl. targets) izgradnje, kot tudi vmesni objekti. Pri gradnji se uporablja t.im. make-datoteka. V njej uporabnik med drugim določi: ciljne objekte, ki naj se zgrade, odvisnosti ciljnih in izpeljanih objektov od drugih izpeljanih objektov in ClearCase-ovih elementov, ter pravila, po katerih naj se te ciljni in izpeljani objekti zgrade.
Značilno za omake je, da lahko sam ugotovi (in v konfiguracijskem zapisu zabeleži) določene odvisnosti med datotekami. Samodejno npr. ugotovi odvisnosti od ustreznih vključitvenih ("include") datotek, ne da bi bile te odvisnosti eksplicitno določene v make-datoteki. Prav tako sam ugotovi odvisnosti od orodij, ki se uporabljajo v pravilih gradnje.
Odgovor na vprašanje, kako omake nadzira izpeljane objekte, se skriva v načinu gradnje in posebnem *mehanizmu pregledovanja* ("wink-in"), ki deluje v dinamičnih pogledih. Orodje omake se loti gradnje danega izpeljanega objekta na precej bolj inteligenten način kot običajni make. Zanj namreč ni pomemben le sistemski datum, temveč cel sklop ClearCase-ovih metapodatkov za dani objekt (za izpeljane objekte so to njihovi konfiguracijski zapisi, za verzije gradnikov pa metapodatki, ki določajo dano verzijo). Gradnje danega objekta se loti v primeru, ko se spremeni vsaj en metapodatek, ki določa kateri koli objekt, od katerega je dani izpeljani objekt neposredno odvisen. Pomembno pri tem je, da se omake pri pregledovanju ne omeji le na dani dinamični pogled, temveč pogleda ("wink-in") v vse dostopne dinamične poglede, ali se morda v njih nahaja že kakšen ažuren izpeljani objekt. Če v določenem dinamičnem pogledu najde tak izpeljani objekt, ga preprosto iz njega potegne v uporabnikovo delovno okolje. Le v primeru, ko v nobenem od pogledov ne najde ustreznega izpeljanega objekta, se loti gradnje tega objekta 'na novo'. Opisani način gradnje v praksi ne le zmanjša možnost napak pri gradnji, temveč posebej pri večjih projektih z mnogimi uporabniki tudi precej prihrani pri času gradnje. Ta učinkovitost se zmanjša v primeru, če uporabniki pri gradnji uporabljajo individualne parametre (npr. opcije prevajalnikov).

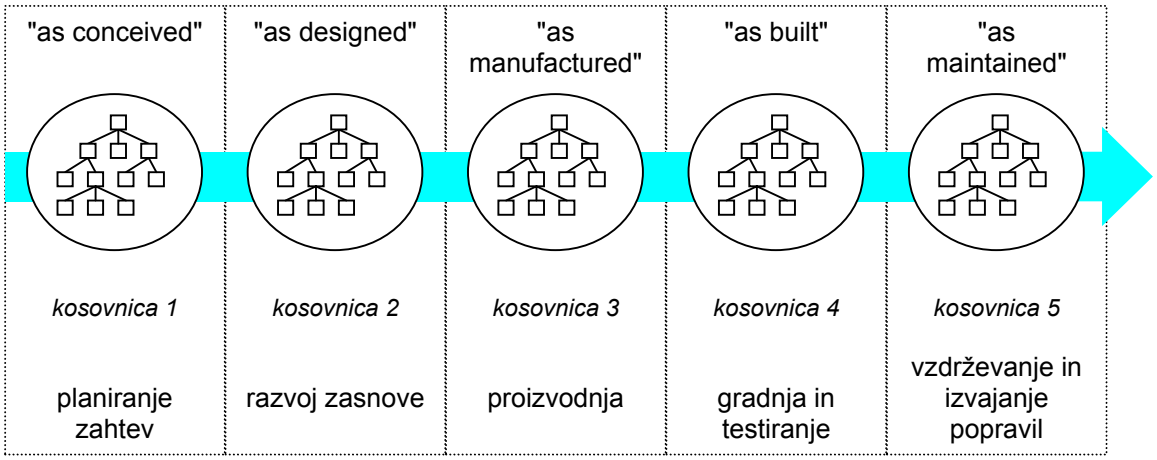
5.2.4 EDS Teamcenter

5.2.4.1 Namen sistema in osnovni koncepti

EDS je eno vodilnih globalnih podjetij na področju sistemske podpore poslovanju. Z razvojem in uporabo novih tehnologij veliko pozornosti vlaga v podporo sodelovanja poslovnih enot in skupin na geografsko porazdeljenih lokacijah. V sodelovanju s svetovalno hišo A.T.Kearney prodaja svoje storitve v 60 državah. Njihov sistem Teamcenter je namenjen podpori upravljanja celotnega življenjskega cikla proizvodov. Omogoča sodelovanje med podjetjem, njegovimi dobavitelji in kupci.

En ključnih konceptov, implementiranih v sistemu Teamcenter, je t.im CPM (krat. Collaborative Product Management). Poslanstvo le tega je v združevanju geografsko razpršenih "otokov znanja o proizvodu" (Aberdeen Group, 2001, str. 2) v enoten vir veljavnih informacij, s katerim bo mogoče na ustrezen način zagotoviti upravljanje celotnega življenjskega cikla produkta. Pri upravljanju procesne verige načrtovanja, izdelave, trženja in vzdrževanja produktov izkorišča informacije iz različnih virov - prodaja, marketing, servisna podpora na terenu, proizvodnja, zagotavljanje kakovosti, dobavitelji komponent izdelka ipd. Posebna pozornost se daje sledljivosti procesa. S tem je močno olajšano izvajanje novih projektov, ki tečejo po enakih vzorcih kot že izvedeni, z dodajanjem novih funkcij že razvitim produktom. Beležijo se zapisi in kosovnice, ki so rezultat posameznih faz življenjskega cikla proizvoda, npr. načrtovanja ("as-designed"), gradnje ("as-built"), vzdrževanja ("as-maintained") ipd. - glej sliko 26.

Slika 26: Podpora upravljanja kosovnic proizvoda s Teamcentrom



Vir: EDS, 2002, str. 3

Konec februarja 2002 je EDS na trg pripeljal novo verzijo sistema - Teamcenter 2.0. Ta se loteva vseh vidikov upravljanja z izdelki. Pri tem se preko svojih modulov povezuje z ostalimi podpornimi in informacijskimi sistemi. Rešitve vključujejo področja: upravljanja zahtev; vodenja projektov in projektnih programov; upravljanja podatkov o proizvodih; povezovanje aplikacij poslovnega sistema; podpora sodelovanja skupin na porazdeljenih lokacijah, pri čemer predstavljajo temelj informacije in podatki o izdelkih.

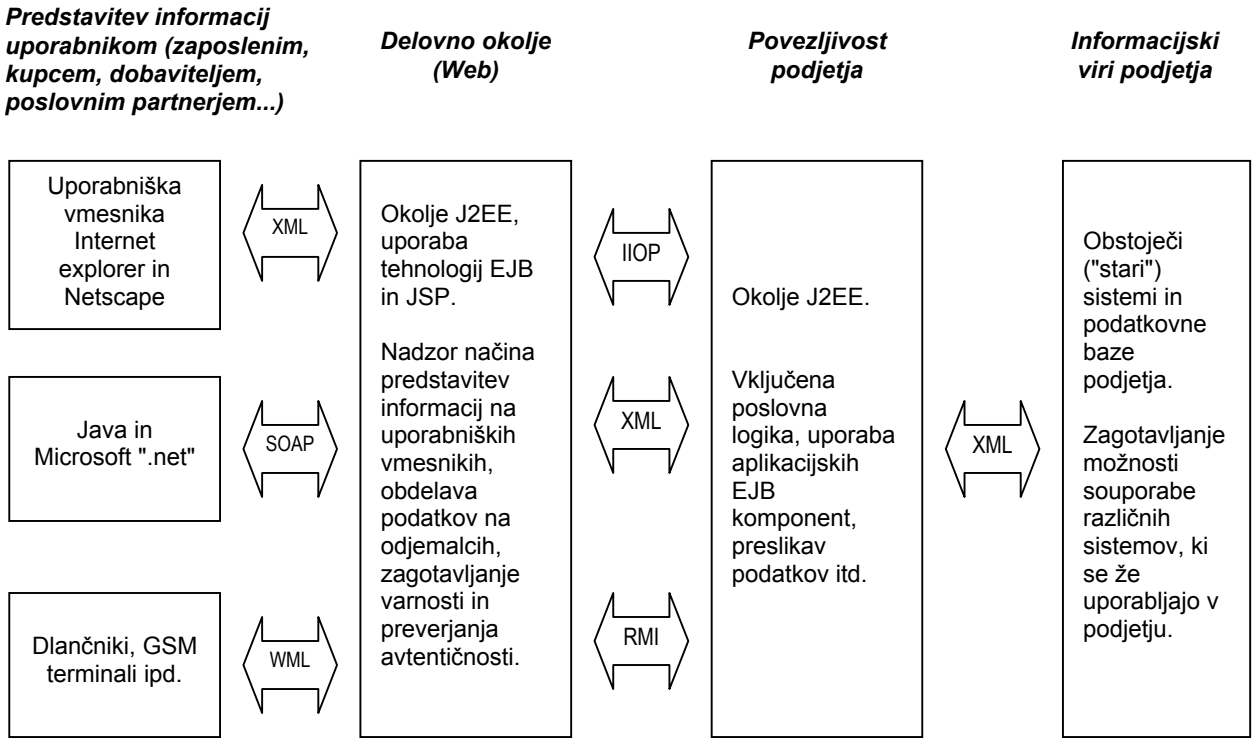
5.2.4.2 Zgradba sistema

Teamcenter je zgrajen okrog skupnega modula *Collaboration Foundation,* ki zagotavlja infrastrukturo in skupne storitve za aplikacije sistema. Usmeritve, ki so bile upoštevane pri snovanju te infrastrukture, so bile: Web okolje, uporaba brskalnikov za doseganje informacij in platforma, ki bo omogočala globalen, neprenehen dostop do podatkov. Strategija v zvezi z razvojem Teamcentra temelji na vključevanju novih modulov, ki bodo podpirali ključne poslovne procese. Temelji zgradbe (Aberdeen Group, 2001, str. 8):

- uporaba standardov skupne uporabnosti (angl. interoperability): XML, SOAP, IIOP, EJB;
- standardi povezljivosti: JNI, RMI, SXML, COM/DCOM, Corba;
- komponente za Web okolje (vključno z vmesniki);
- možnost povezovanja z drugimi poslovnimi sistemi preko Web strežnikov;
- možnost povezovanja in skupnega delovanja z različnimi PDM sistemi, s pomočjo aplikacijskih vmesnikov;

- uporabniški vmesnik, katerega izgled sledi izgledu Microsoftovega namizja. Teamcenter podpira 'tanek' (angl. thin) brskalnik za Web z namenom zagotavljanja splošne dosegljivosti, Javine odjemalce za grafične aplikacije in uporabo active-x (tipa strežnik/odjemalec) za nadzor interakcij pri delu z uporabniškimi aplikacijami;
- razdelitev zgradbe Teamcentra na različne komponente omogoča lažje in ažurnejše vključevanje novih tehnologij in inovacij.

Slika 27: Zasnova sistema Teamcenter



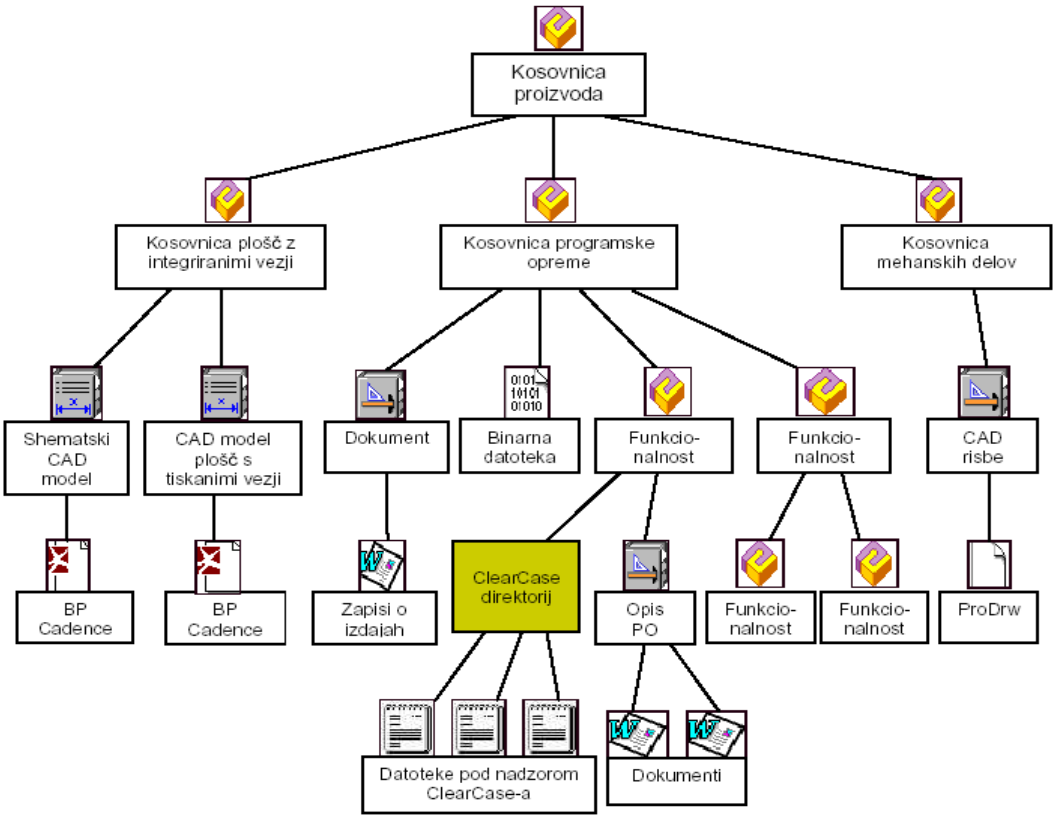
Vir: Aberdeen Group, 2001, str. 8

5.2.4.3 Funkcionalnosti sistema

Teamcenter podpira naslednje pomembnejše funkcije v zvezi z upravljanjem konfiguracij in sprememb izdelkov (EDS, 2002a, str. 9):

- upravljanje odjav/prijav (check-in/check-out);
- upravljanje z revizijami in vmesnimi verzijami nadzorovanih objektov oziroma gradnikov;
- upravljanje s konfiguracijo proizvodov in njihovih komponent, s pomočjo sestavnic, kosovnic in zbirk (vključeno tudi razvrščanje teh objektov);
- upravljanje procesa, posebej na področju inženirskega procesa spreminjanja izdelkov;
- upravljanje z dokumentacijo za proizvode;
- vložena vizualizacija - možnost pregledovanja rezultatov načrtovanja, izdelanih z različnimi CAD orodji, v dveh ali treh dimenzijah;
- globalno doseganje do podatkov s katere koli lokacije v porazdeljenem poslovnem okolju.

Slika 28: Organizacija kosovnic proizvodov in možnosti vključevanja PO



Vir: Interna dokumentacija podjetja (korespondenca z EDS)

Vodenje konfiguracij in sprememb v Teamcentru je podprto s procesnim modelom ICM CMII, (Marshall, 2002, str. 3). V Teamcenter je v določeni meri vključena PDM funkcionalnost sistema Metaphase. Ena od lastnosti, ki jo je Teamcenter podedoval od tega sistema, je možnost povezovanja s sistemom ClearCase. Osnovne značilnosti povezovanja med omenjenima sistemoma (EDM/PDM in SCM):

- EDM/PDM sistem lahko vsebuje informacije o gradnikih in objektih, ki se nahajajo izven tega sistema. Primer: sistem ima le informacijo o kosovnici, same kosovnice pa dejansko ne vsebuje oziroma shranjuje. To pomeni, da ko v EDM/PDM sistemu kreiramo kosovnico, kreiramo le informacijo o tej kosovnici. Enak koncept velja tudi za druge objekte konfiguracije. Datotečni sistem v Metaphase-u je pravzaprav le lokacija dejanskega datotečnega sistema - določena je s sistemom gostiteljskega računalnika in potjo datotečne mape na tem sistemu. Ko kreiramo datotečni sistem v EDM/PDM sistemu, se gostiteljski računalnik in lokacija shranita v sistemu, pri tem pa se datotečni sistem zunaj EDM/PDM sistema ne kreira. Podobno velja za odlagališča gradnikov: lokacije delovnih okolij in odlagališč gradnikov so v okviru EDM/PDM sistema kazalci do map, vsebovanih v datotečnih sistemih, definiranih v EDM/PDM sistemu;
- ko so datotečni sistem, delovno okolje in odlagališče podatkov v Metaphase-u kreirani, uporabniki lahko dosegajo vsebino teh lokacij. Na samem začetku se tako datoteka kot njena vsebina v ustrezni datotečni mapi obravnava kot t.im. "online" podatek, saj v podatkovni bazi EDM/PDM sistema še ni shranjenih nobenih informacij (povezav) o teh objektih. Procesu shranjevanja povezav do teh objektov pravimo *registracija*. Zato "online" objektom pravimo tudi neregistrirani gradniki. V EDM/PDM sistemu se lahko odpirajo in pregledujejo tudi neregistrirane datotečne mape. Datoteke se lahko

pregledujejo in izvajajo. Za izvajanje operacij v zvezi z upravljanjem konfiguracij (odjavljanje, povezovanje z kosovnicami ipd.) pa mora biti dani objekt registriran;

- registrirani podatkovni gradniki so tisti podatkovni gradniki, o katerih ima EDM/PDM sistem shranjene metapodatke. Kot rečeno, EDM/PDM sistem ne shranjuje samih gradnikov, temveč le podatke o gradnikih (metapodatke). Informacija, ki jo hrani EDM/PDM sistem, se lahko povezuje z drugimi gradniki v življenjskem ciklu izdelka, lahko se spreminja njeno lastništvo, itd.. Med drugim se shranjuje in vodi tip gradnika, s katerim se določa način njegove obdelave.

5.2.4.4 Proces sprememb v Teamcentru

Proces spreminjanja izdelkov je en ključnih procesov, ki vpliva na spreminjanje konfiguracij proizvodov. Uporabniške vloge v tem procesu so:

- lastnik (angl. owner) - avtor gradnika ali odgovorni za njegovo spremembo;
- podajalec zahteve (angl. requestor) - odgovorni za podajanje in izpolnitev spremembe;
- administrator - upravlja s spremembo na najvišjem nivoju;
- analitik (angl. analyst) - preverja tehnični vidik spremembe in izvede spremembo;
- odbor za pregled (angl. review board) - pregleda, potrdi in avtorizira spremembo.

V procesu sprememb nastopajo naslednji gradniki oziroma objekti (Marshall, 2002, str. 6):

- *poročilo o problemu (angl. problem report, krat. PR)* sproži spremembo. Njegov namen je jasno določiti problem in zaporedje dogodkov, ki je do tega problema pripeljalo. Služi kot osnova za podajo zahteve za spremembo;
- *zahteva za spremembo (angl. change request, krat. CR)* je običajno odgovor na poročilo o problemu in lahko vsebuje delovni načrt, sestavljen iz strategij in/ali nalog, ki povedo kaj in kako naj se spremeni. Vključuje tako oceno stroškov kot koristi v zvezi z izvedbo spremembe. V njej so lahko poistoveteni gradniki izdelka, na katere dana sprememba učinkuje. Predstavlja temelj za poslovno odločanje v zvezi z izvedbo spremembe;
- *obvestilo o spremembi (angl. change notice, krat. CN)* ima hkrati vlogo delovnega naloga. Vpeljuje podrobni delovni plan (načrt) v zvezi z definiranimi strategijami in nalogami. V njem je natančno določeno, kaj naj se spremeni in na kakšen način. V njem so lahko identificirani gradniki izdelka, na katere dana sprememba učinkuje. Poleg tega so lahko določeni sinhronizacijski datumi, datumi učinkovanja spremembe itd. S pomočjo obvestila o spremembi se odobrijo in dodelijo aktivnosti v zvezi s spremembo;
- *strategija (angl. strategy)* določa delo, potrebno za izvedbo spremembe, na najvišjem nivoju. Na dano poročilo o problemu, zahtevo za spremembo in obvestilo o spremembi se lahko veže več strategij. Strategija je lahko sestavljena iz drugih strategij in/ali nalog. Uporaba strategije v procesu sprememb ni neobhodna;
- *naloga (angl. task)* določa podrobnosti v zvezi z izvedbo spremembe in se nanaša na posamezne akcije. Naloga se običajno veže na določeno strategijo. Na dano poročilo o problemu, zahtevo za spremembo in obvestilo o spremembi se lahko veže več nalog. Tudi določanje nalog se v procesu sprememb lahko zaobide.

Poleg tega nastopajo še povezave med omenjenimi gradniki in objekt *učinkovanje*. Ta objekt se vodi neodvisno od same spremembe. Z njim je npr. lahko določen datum, kdaj sprememba začne učinkovati (t.im. *datumsko učinkovanje*), omogoča pa tudi vodenje drugih podatkov v zvezi z učinkovanjem sprememb.

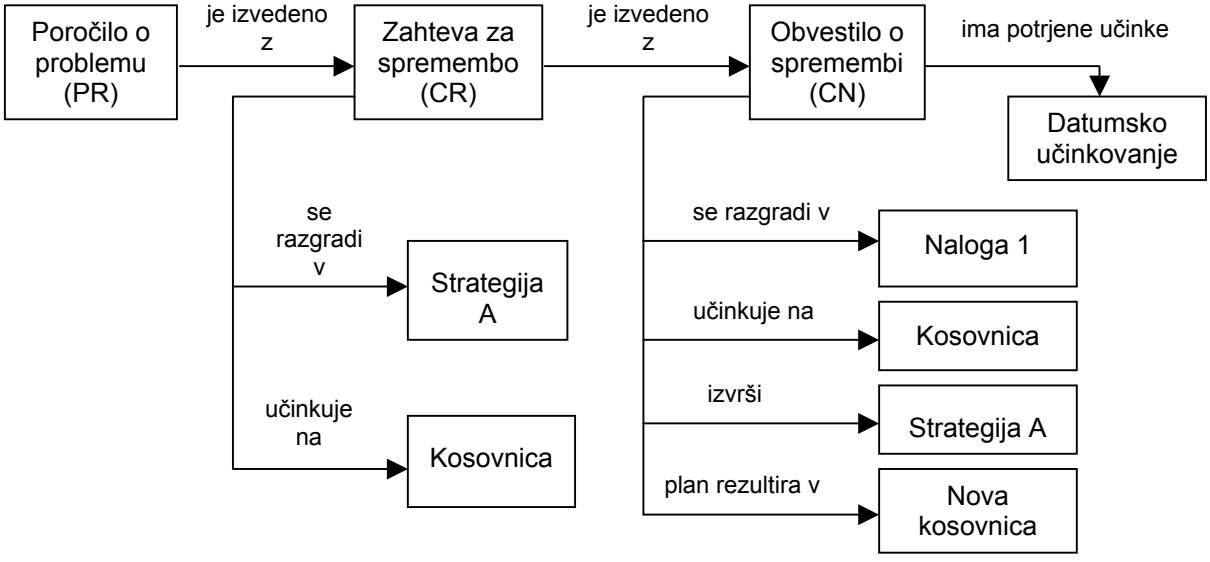
Proces sprememb je določen z naslednjimi aktivnostmi:

- *poistovetenje (identifikacija) in potrditev problema.* Najprej lastnik ali podajalec zahteve kreira poročilo o problemu in določi zaporedje dogodkov, ki je do pojavitve problema

pripeljalo. Določi se administrator za ta problem. Po potrditvi problema administrator preda spremembo analitiku, ki bo spremembo pregledal. Analitik pregleda problem in potrdi, ali gre dejansko za problem. Dokumentira rezultate pregleda;

- *določitev in odobritev zahteve za spremembo.* Razvijalec ali podajalec zahteve kreira zahtevo za spremembo. Oceni stroške in koristi v zvezi s spremembo ter priporočene akcije za reševanje problema. Določi prizadete objekte oziroma gradnike proizvoda in določi strategijo za izvedbo spremembe, ki jo poveže s poročilom o problemu. Spremembo preda nazaj administratorju. Sledi obravnava spremembe. Administrator preda zahtevo za spremembo odboru za pregled (angl. review board). Odbor pregleda zahtevo za spremembo in potrdi predlagano rešitev (v primeru zavrnitve se cikel ponovi);
- *določitev in odobritev obvestila o spremembi.* Razvijalec ali podajalec zahteve kreira obvestilo o spremembi in ga poveže s prizadetimi gradniki oziroma kosovnicami proizvoda. Nato določi nalogo, potrebno za izvedbo spremembe. Obvestilo o spremembi nato poveže s predhodno določeno strategijo in zahtevo za spremembo. Določi, kdaj bo sprememba začela učinkovati. Obvestilo o spremembi poveže tudi z objektom učinkovanje. Sledi faza odobravanja obvestila o spremembi. Administrator določi analitika, ki bo izvedel spremembo. Nato odbor za pregled potrdi plan izvedbe;
- izvedba spremembe. Administrator preko sistema spremlja izvajanje spremembe. Analitik pri tem izvede spremembe v skladu s podrobnim izvedbenim planom;
- zapiranje spremembe. Ko analitik izvede spremembo, na ustrezen način spremeni stanje življenjskega cikla spremembe in le-to preda administratorju. Ko so vse aktivnosti v izvedbenem planu opravljene, ta zapre spremembo na ustreznem nivoju, od spodaj navzgor: naloga, strategija, obvestilo o spremembi. Ko je zaprt tudi slednji (t.j. CN), se samodejno zapro tudi zahteva za spremembo, ustrezna strategija in poročilo o problemu.

Slika 29: Povezave med gradniki in objekti procesa sprememb v Teamcentru



Vir: Marshall, 2002, str. 30

5.2.5 mySAP PLM

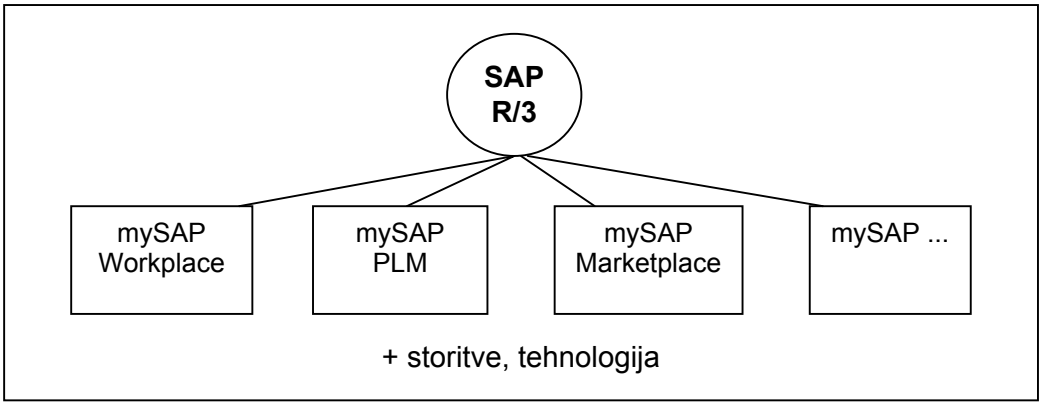
5.2.5.1 Poslovni sistem SAP in njegov pristop k upravljanju življenjskega cikla produktov

Podjetje SAP je nastalo leta 1972. Je eno najbolj poznanih in priznanih svetovnih dobaviteljev rešitev s področja elektronskega poslovanja. Program mySAP PLM je bil spočet leta 1996 in od tedaj naprej podpora upravljanja podatkov o produktih predstavlja eno od strateških usmeritev podjetja. SAP si je skozi leta svojega poslovanja pridobil predvsem izkušnje na področju podpore proizvodnje industrijskih izdelkov, ima pa razmeroma malo izkušenj na področju inženirskega načrtovanja (CIMdata, 2001, str. 6).
Funkcije modula mySAP PLM so namenjenu podpori upravljanju življenjskega cikla proizvodov (CIMdata, 2001, str. 6). V ta sistem so vgrajena naslednja pravila dobre prakse (angl. best practices) in tehnologije: PDM (krat. za angl. Product Data Management); podpora sodelovanja in skupinskega dela; vizualizacija; CSM (krat. za angl. Component Supplier Management) ipd.

5.2.5.2 Zgradba sistema, načini povezovanja in prenosa podatkov

Zgradba sistema mySAP PLM Verzija 4.6C je porazdeljena (CIMdata, 2001, str. 42) in trislojna. Sloji so ločeni na podatkovno bazo, aplikacijo in predstavitveni sloj. Podatki, ki jih nadzoruje SAP PLM, se lahko hranijo na kateri koli od porazdeljenih lokacij, ki jih sistem nadzoruje. Podatke je možno nadzorovati z enega strežnika ali s pomočjo porazdeljene množice strežnikov. Strežnika, namenjena aplikaciji in podatkovni bazi, lahko tečeta na različnih operacijskih sistemih. Pri tem mySAP PLM podpira različne sisteme za upravljanje relacijskih podatkovnih baz (RDBMS), kot npr. Oracle, DB2, Microsoft SQL Server, Informix in SAP DB.

Slika 30: Komponente in storitve mySAP.com

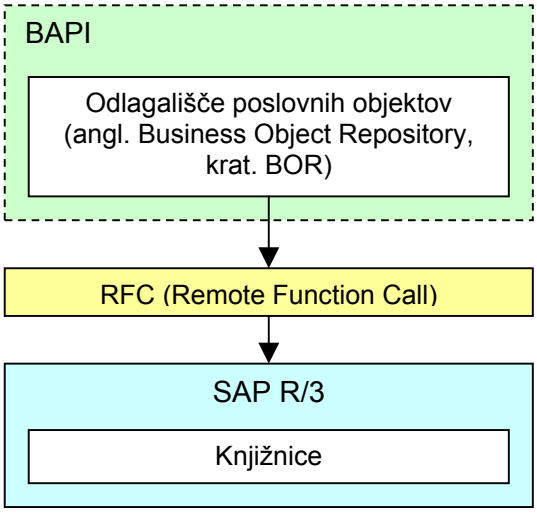


Vir: Kežmah, Heričko, 2001, str. 187

mySAP PLM je zgrajen s SAP-ovim jezikom četrte generacije, ABAP. Pri tem so uporabljene objektno naravnane razširitve ABAP-ovih objektov. Pravkar omenjena SAP-ova razvojna platforma je na voljo tudi uporabnikom sistema za izvajanje prilagoditev lastnim zahtevam. Številni dejavniki, med njimi npr. razširjenost SAP-ovih sistemov po svetu in sam obseg rešitev, ki jih ponuja SAP, silijo podjetje SAP v izjemno pazljivost pri načrtovanju zgradbe sistemov in uporabljenih pristopov pri načrtovanju rešitev. Le z zelo sistematičnim pristopom in učinkovitim obvladovanjem zgradbe je namreč mogoče zagotoviti usklajeno delovanje celotnega sestavljenega sistema in preprečiti pojavitev nekonsistentnosti v sistemu. Večina SAP-ovih novejših rešitev in modulov je objektno zasnovanih. Sistemi se gradijo in širijo okrog osnove, ki jo predstavljata operacijski sistem in baza podatkov.

V sistemu mySAP je veliko pozornosti posvečeno povezovanju in prenosu podatkov (Robinson, 2002). Sistem za izmenjavo podatkov uporablja mehanizem BAPI-jev (angl. Business Application Programming Interface) oziroma vmesnikov za programiranje poslovnih aplikacij. Že v osnovni namestitvi SAP R/3 je na voljo več sto BAPI-jev za podporo različnih področij poslovanja (računovodstvo, logistika, upravljanje s človeškimi viri itd.). Ti lahko predstavljajo poslovne objekte, transakcijske zapise ipd. in so namenjeni izmenjavi podatkov med sistemom SAP in drugimi sistemi za posporo poslovanju. BAPI-ji pri opravljanju svojih operacij prenosa podatkov uporabljajo mehanizem RFC (Remote Function Call). RFC je funkcionalna koda oziroma funkcijski modul sistema, ki se lahko kliče iz drugega sistema (Robinson, 2002). Ta je lahko ali SAP-ov modul ali kakršen koli drug sistem, ki zna obvladovati omenjeni mehanizem.
BAPI temelji na uporabi SAP-ovega odlagališča poslovnih objektov, BOR (krat. za Business Object Repository). Vsebina BOR-a v bistvu določa poslovni model podjetja. Vključene informacije določajo tako poslovne objekte, ki se uporabljajo znotraj podjetja, kot tudi vse vmesnike proti zunanjim podjetjem oziroma poslovnim partnerjem.
Za prenos in izmenjavo podatkov SAP uporablja tudi mehanizem IDoc, ki se uporablja predvsem za izmenjavo informacij med posameznimi moduli te aplikacije.

Slika 31: Uporaba mehanizmov BAPI in RFC



Vir: Robinson, 2002

SAP PLM ima vmesnike do aplikacij za razvoj mehanskih konstrukcij (npr. SDRC IDEAS) in razvoj elektronike oziroma integriranih vezij (npr. Mentor Graphics). Ti aplikaciji imata svoje lastne module za upravljanje konfiguracij oziroma kosovnic komponent. Ko se komponenta razvije do določenega mejnika, se njena izhodiščna konfiguracija preseli pod nadzor PDM ali PLM sistema, kjer se vodijo informacije o celotnem proizvodu.
SAP premore določene storitve in operacije tudi na internetu, pri čemer se večinoma uporabljajo standardni programski jeziki za Web. Nekateri SAP-ovi moduli npr. uporabljajo Java IDE za izvajanje prilagoditev na uporabniških vmesnikih za Web. Za izdelavo aplikacij v tem (t.j. Web) okolju se uporablja tudi razvojno okolje Web Application Server. To je odprto okolje, ki podpira različne standarde za internet, npr. HTTP(s), XML, SOAP, dobro pa se ujema tudi s programskimi koncepti, ki predvidevajo uporabo skriptov na strežnikih.

5.2.5.3 Funkcije sistema

mySAP.com podpira vse objekte, ki pridejo v poštev pri upravljanju podatkov o proizvodih in projektih: uradne gradnike, vključno z dokumentacijo, sestavnice (BOM), strukture projektov ipd.

Zagotovljen je vsestranski opis izdelkov s konsistentno konfiguracijo, dokumentacijo, izhodiščnimi konfiguracijami. V različnih fazah celotnega življenjskega cikla proizvodov se upravljajo različni tipi konfiguracij in z njimi povezane funkcije, npr.: konfiguracije načrtov proizvodov ("as engineered"); konfiguracije, določene za posamezna prodajna naročila ("as sold"); konfiguracije, določene pri gradnji posameznega proizvoda ("as build"), ali že dostavljeni proizvodi, ki so zdaj v fazi vzdrževanja ("as maintained"). SAP podpira naslednje osnovne funkcije, vgrajene v cPDM rešitev (CIMdata, 2001, str. 16):

- upravljanje odlagališča podatkov in upravljanje dokumentacije;
- upravljanje pretoka dela (angl. workflow) in procesov;
- upravljanje strukture proizvodov, katerega del je tudi upravljanje konfiguracij proizvodov;
- upravljanje razvrščanja (klasifikacije);
- upravljanje projektov in programov (družine podobnih projektov).

Upravljanje s strukturo izdelkov se deli naprej na: upravljanje s sestavnicami (BOM); upravljanje konfiguracij; povezano inženirstvo izdelkov in procesa; konfigurator za internet. V nadaljevanju je predstavljeno nekaj z obravnavanega zornega kota zanimivejših funkcij.

5.2.5.3.1 Upravljanje podatkovnega skladišča in upravljanje dokumentacije

Tipe dokumentov oziroma gradnikov lahko določi uporabnik sam. Pri tem mySAP PLM za vsako verzijo dokumenta kreira in vzdržuje svoj zapis. Uporabnik lahko dokumente poveže z drugimi objekti sistema. V tem trenutku je takšnih objektov, ki jih je mogoče uporabljati v zvezi z obvladovanjem dokumentacije, preko 40. Dokument se lahko npr. neposredno poveže z enoto dela, ki se uporablja pri razgradnji projektov (angl. work breakdown structure, krat. WBS).
Funkcionalnost upravljanja dokumentacije je vključena v sistem za podporo pretoka dela. Ta na avtomatiziran način podpira razdelitev in upravljanje nalog, povezanih z upravljanjem dokumentacije. Vseskozi se beleži vsaka sprememba stanja dokumenta, pri čemer se zapišejo še informacije o uporabniku, času in datumu, povezanimi z omenjeno spremembo stanja. Spremembe stanj se lahko povežejo tudi s sprožilci, ki ob spremembi izvedejo ustrezne akcije oziroma operacije. Z dokumentacijo med drugim lahko izvajamo naslednje operacije:

- razvrščanje dokumentov v različne skupine, razrede in tipe;
- iskanje na podlagi lastnosti, ki jih ima dani razred;
- urejanje dokumentacije v hierarhične ali nehierarhične strukture. Pri tem se zagotavlja podpora *sestavnic dokumentacije* (angl. bill of documents, krat. BOD).

V okviru logičnega odlagališča podatkov, ki ga nadzira in podpira sistem, lahko določimo različna fizična odlagališča podatkov v računalniški mreži. Če uporabnik dela z gradniki, ki so vključeni v različna odlagališča, potem se ob vsaki prijavi (operacija check-in) gradnika v odlagališče specificira izbrano odlagališče. Sistem pri tem podpira različne nivoje dodeljevanja dovoljenj v zvezi z doseganjem gradnikov, ki so pod njegovim nadzorom. Dovoljenja lahko dodelimo posameznim uporabnikom, skupinam uporabnikov ali vlogam v procesu.

5.2.5.3.2 Upravljanje konfiguracij in sprememb proizvodov

SAP podpira vsa štiri funkcionalna področja upravljanja konfiguracij: identificiranje konfiguracij; nadzor konfiguracij; vodenje stanj konfiguracij in presojanje konfiguracij. Vodijo se povezave med gradniki, proizvodi, dokumentacijo, spremembami, aktivnostmi in viri (preko izhodiščnih konfiguracij) skozi celotni življenjski cikel izdelka. Z nadzorovanjem vseh sprememb nad konfiguracijo proizvoda se zagotavlja konsistentnost samega izdelka. Za vodenje informacij o izhodiščnih konfiguracijah mySAP PLM uporablja t.im. *konfiguracijsko mapo* (angl. configuration folder). Znotraj te mape se določajo in zbirajo gradniki. Seznami teh gradnikov določajo proizvod. Poseben objekt, *oglavje mape* (angl. folder header), povezuje izhodiščne konfiguracije

z drugimi objekti in moduli za vodenje sprememb proizvodov v mySAP PLM, med drugim tudi s sistemom za vodenje dokumentacije.
Koncept *ocene inženirske spremembe* (angl. Engineering Change Notice, krat. ECN) je ključen za uspešno podporo procesa upravljanja sprememb z mySAP PLM. ECN se uporablja za vzdrževanje povezav z drugimi objekti v zvezi s spremembo, npr. dokumenti in kosovnicami. Z ECN-om uporabnik določi spremembo na produktu in aktivnosti, potrebne za izvedbo spremembe. Potrjeni ECR se pretvori v ECO, na podlagi katerega se izvedejo operativne spremembe gradnikov, vključno z njhovimi potrditvami in izdelavo novih revizij.

5.2.6 Telelogic CM Synergy

5.2.6.1 Predstavitev osnovnih konceptov, vgrajenih v CM Synergy

Švedsko podjetje Telelogic, ustanovljeno l. 1983, je globalni dobavitelj programskih rešitev. Usmerja se v telekomunikacijsko, letalsko, obrambno, avtomobilsko in druge industrije. Največje izkušnje imajo z aplikacijami s področja inženiringa, informacijskih tehnologij in finančne podpore (Taborda, 2001, str. 3). Telelogic v zvezi s podporo življenjskega cikla PO ponuja module, kot so CM Synergy za upravljanje konfiguracij, Change Synergy za upravljanje sprememb, DocExpress za upravljanje dokumentacije in izdelavo poročil, DOORS za upravljanje življenjskega cikla zahtev za sisteme, UML Suite za modeliranje zahtev, SDL Suite za razvoj sistemov, Logiscope za testiranje PO. Modul CM Synergy, namenjen upravljanju s konfiguracijo PO, temelji na dveh konceptih: podpori komponentnemu načinu razvoja programske opreme in na nalogah temelječem upravljanju konfiguracij in sprememb.

5.2.6.1.1 Podpora komponentnemu načinu razvoja PO

CM Synergy, modul za upravljanje s konfiguracijo PO, podpira *komponentni način razvoja* (angl. component-based development), t.j. ustvarjanje sistemov iz pouporabljivih komponent. Pri tem je tu pod pojmom komponenta mišljena kakršna koli zaključena enota PO z določenimi, specificiranimi odvisnostmi in vmesniki proti drugim komponentam sistema. V industriji je opazen porast tovrstnega načina razvoja programskih sistemov, katerega prednost sta nižji razvojni stroški in krajši razvojni časi. Znanost na tem področju še ni dosegla zrelosti. Trenutno se izvaja večje število raziskav (Wiborg Weber, Vignaud, 2001, str. 3), ki se ukvarjajo z izzivi in še nerešenimi problemi, predvsem na področjih zagotavljanja konsistentnih verzij komponent pri njihovem sestavljanju v končni izdelek ter nalogami v zvezi z zagotavljanjem sledljivosti in možnostmi uporabe komponent v geografsko porazdeljenem razvojnem sistemu.

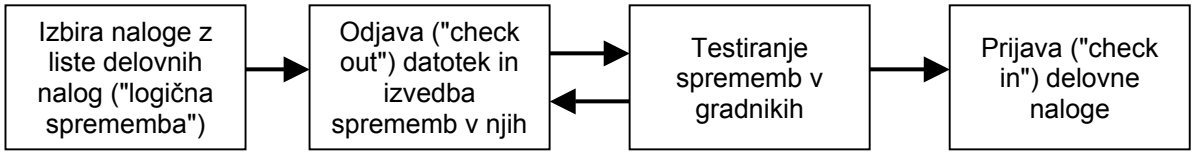
5.2.6.1.2 Na nalogah temelječ proces upravljanja konfiguracij in sprememb

CM Synergy podpira v naloge usmerjen proces upravljanja konfiguracij. To pomeni, da se dana sprememba na komponenti navezuje na podano nalogo, ne na posamezne datoteke, gradnike. Uporabnik dela z nalogami s pomočjo seznama, s katerega izbere tekočo nalogo (Taborda, 2001, str. 37). Sistem nato samodejno v delovnem področju nadzoruje množico gradnikov, ki jih uporabnik spreminja v okviru dane naloge. Nekaj prednosti v naloge usmerjenega upravljanja konfiguracij (Wiborg Weber, Vignaud, 2001, str. 3):

- intuitivnost. Razvijalci lahko dano nalogo poistovetijo z logično spremembo, ki jo nato preslikajo na množico spremenjenih komponent oziroma gradnikov. Primer: Vse spremembe, povezane z nalogo, se npr. prijavljajo (z operacijo 'check in') v odlagališče skupaj, kot povezana množica. Celoten cikel je v grobem prikazan na sliki 32);

- učinkovitejše obvladovanje konfiguracij. Konfiguracijo lahko obravnavamo kot osnovnico oziroma izhodiščno konfiguracijo, ki ji dodamo množico pravkar zaključenih ali izvajajočih se nalog oziroma logičnih sprememb;
- možnost povezave z zahtevami za spremembe. Nalogo lahko kreiramo za določeno zahtevo za spremembo, s čemer je zagotovljena učinkovita povezava med problemi na izdelku oziroma novimi funkconalnostmi izdelka in dejanskimi spremembami v programskem izdelku;
- proces je mogoče prilagoditi različnim zahtevam. Lahko se vodijo različni tipi procesov za različne tipe izdaj (npr. uradne, testne...) PO ali za različne razvojne skupine: od manj formalnih, npr. prototipnih in spiralnih modelov, do bolj formalnih (npr. model vodnega slapu).

Slika 32: Upravljanje sprememb na temelju nalog



Vir: Taborda, 2001, str. 41

5.2.6.2 Zgradba CM Synergy

Predstavitev temelji na verziji CM Synergy 5.1 (junij 2000) in kasnejših izvedbah. Sistem izhaja iz sistema Continuus CM ameriškega proizvajalca Continuus Software, ki je na tržišču SCM sistemov že od leta 1991. Od tega sistema je podedoval tudi svojo zasnovo in funkcionalnosti.
CM Synergy ima zgradbo tipa strežnik/odjemalec (Yphise, 2000, str. 31). Kot pri vseh SCM sistemih tudi pri sistemu CM Synergy temelj predstavlja odlagališče podatkov in metapodatkov (v tem primeru imenovano tudi: *odlagališče komponent*). V njem so enolično poistovetene komponente PO, pri čemer se za vsako od njih vodijo verzije in drugi metapodatki (Wiborg Weber, Vignaud, 2001, str. 6). Za komponente se vodijo tudi povezave do drugih komponent. Podprto je upravljanje tako s komponentami tipa "črnih škatel" (angl. black box), npr. knjižnicami, kot s t.im. izvornimi komponentami, za katere se vodi njihova drevesna struktura. Za vsako komponento se lahko hranijo dodatni gradniki oziroma dokumentacija, vključujoč zahteve, specifikacije, testne vzorce, rezultate testiranj. Pri tem se pazi, da so verzije teh gradnikov vseskozi usklajene z verzijami samih komponent. Metodologija, vgrajena v CM Synergy, omogoča upravljanje različnih variant (različic) za dano komponento, izvajanje vzporednega razvoja z uporabo različnih razvojnih (oziroma 'izdajalnih') tokov ter upravljanje in združevanje vzporednih sprememb v okviru teh tokov. Omogočena je uporaba porazdeljenih odlagališč v geografsko porazdeljenem razvojnem okolju, pri čemer se komponente porazdele med ta odlagališča na najbolj primeren način.
V zadnjih letih je sistem doživel precej sprememb. Izboljšani so bili uporabniški vmesniki, izboljšana je bila razvojna podpora na platformah Unix in Windows. Trenutno CM Synergy podpira naslednje pomembnejše platforme:

- za strežnik: Unix (HP-UX, Linux, Sun Solaris, AIX itd.), Windows (NT, 2000, 9x);
- za odjemalce: Unix (AIX, Digital Unix, HP-UX, Linux, Sun Solaris, Irix, Sinix), Windows (NT, 2000, 9x), vmesnik za Web (Web Synergy ali Change Synergy, z obdelavo zahtev za spremembe PO).

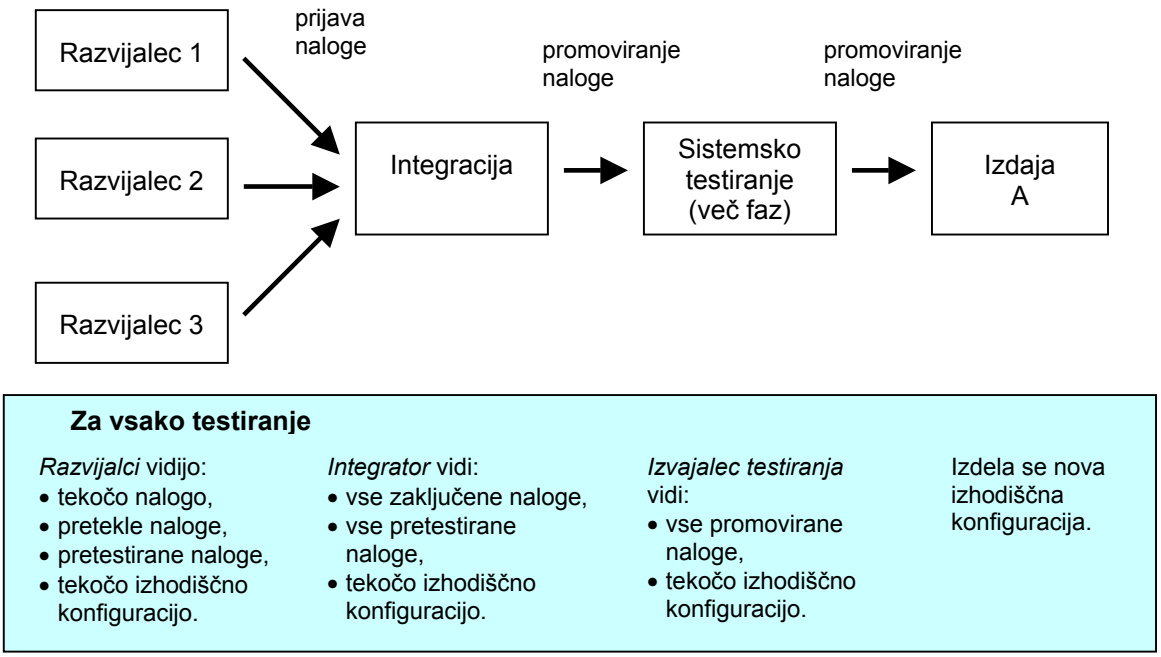
Od zanimivejših vmesnikov do drugih aplikacij za podporo življenjskega cikla PO ponuja vmesnik do Rationalovega modula Rational Rose, namenjenega modeliranju zahtev za programske sisteme.

5.2.6.3 Funkcije sistema CM Synergy

CM Synergy ponuja okvir in proces za upravljanje s konfiguracijami komponent v geografsko porazdeljenem okolju. Ta okvir vsebuje (Wiborg Weber, Vignaud, 2001, str. 6):

- odlagališče komponent in metapodatkov - opisano v prejšnjem razdelku;
- upravljanje s konfiguracijami komponent oziroma sistemov. CM Synergy uporablja komponente kot gradbene bloke za razvoj drugih sistemov. Komponente so lahko sestavljene iz drugih komponent oziroma gradnikov; lahko se jih uporablja tudi kot posamezne gradnike. Podprta je tudi obratna pot, t.j. razgradnja komponent na sestavne dele, na podlagi česar je izvedeno verzioniranje komponent, podpora paralelnemu razvoju in pouporaba posameznih sestavnih delov. Sistem ima možnost učinkovitega sledenja in iskanja, kje se določena verzija komponente uporablja;
- podpora procesu, temelječem na nalogah. Tipičen potek procesa: naloga se dodeli razvijalcu; ko razvijalec dela na nalogi, se spremembe, ki jih izvede, samodejno povežejo z nalogo; ko so spremembe opravljene, razvijalec označi nalogo kot zaključeno. Naloga tako združuje spremembe na gradnikih v enotno logično spremembo, ki nato potuje skozi svoj življenjski cikel. CM Synergy podpira različne vloge v procesu (Wiborg Weber, Vignaud, 2001, str. 14), s katerimi so povezani mehanizmi v zvezi s pretokom dela in zagotavljanjem varnosti: *developer* - za uporabnike, ki razvijajo PO; *component_developer* - za uporabnike, ki razvijajo PO in izdajajo komponente; *build_mgr* - za uporabnike, ki zbirajo in sestavljajo PO, npr. z namenom izvajanja integracijskega ali sistemskega testiranja. Pri tem CM Synergy podpira povezavo med obravnavo nalog in projektnimi aktivnostmi, vodenimi v aplikaciji Microsoft Project;

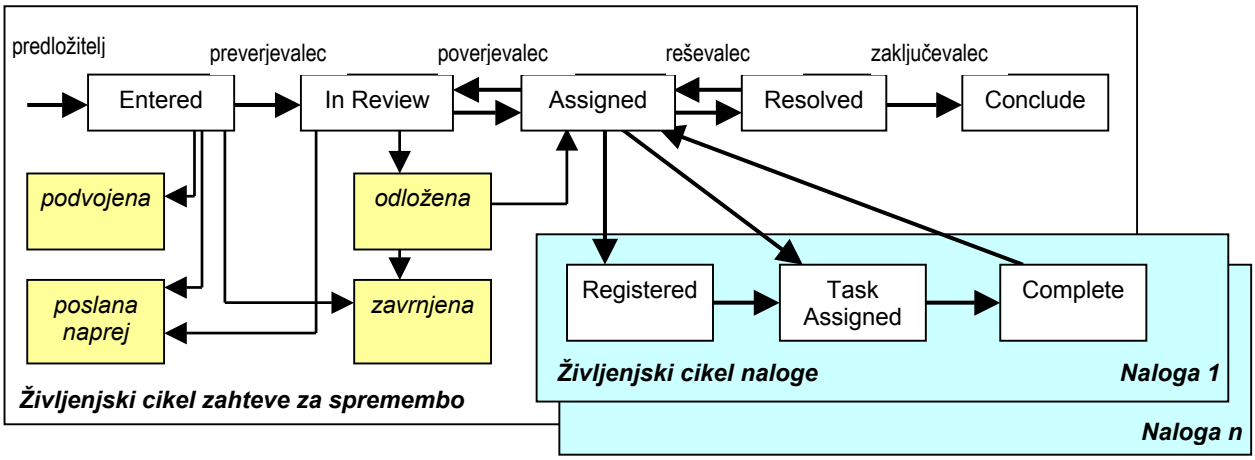
Slika 33: Razvojna dinamika



Vir: Taborda, 2001, str. 43

- podpora delovnih okolij, za razvoj in testiranje. Omogočena je uporaba ločenih delovnih okolij za razvijalce, prav tako pa tudi način dela, kjer več razvijalcev dela na isti komponenti. Ločena okolja razvijalcem omogočajo nemoteno delo, ne da bi sami motili druge ali bili moteni od drugih, vse dokler njihovo delo ni dokončano. Po zaključku spremembe razvijalec ažurira svoje okolje s spremembami drugih razvijalcev in preveri svoje spremembe še v kontekstu sprememb drugih. Poleg razvojnih se lahko uporabljajo tudi druga delovna okolja, namenjena izvajanju posameznih faz testiranja, npr. za integracijsko testiranje, sistemsko testiranje ipd.;
- podpora izdajanju PO. Izdaje PO temelje na nalogah (Taborda, 2001, str. 39). Naloge gredo skozi različna stanja v svojem življenjskem ciklu. Pri tem se verzije različnih objektov, ki jih sistem podpira, navezujejo na naloge. Dodajanje nalog ali njihovo opuščanje je razmeroma enostavno, podprto je izvajanje različnih analiz (npr. tipa "KAJ-ČE"). Z zgodnjim zaznavanjem konfliktnih situacij (npr. nenamernih odvisnosti med gradniki) je preprečeno izvajanje nepotrebnih gradenj programskih paketov. Pravila določanja konfiguracije, ki se uporablja pri izdajanju PO, temelje na uporabi izhodiščne konfiguracije in izvedenih nalog, planiranih za novo izdajo;
- upravljanje sprememb na komponentah in njihovih združevanj. Z uporabo dodatnega modula ChangeSynergy je podprt mehanizem za sledenje in upravljanje inženirskih sprememb na programskem sistemu. Pri obdelavi zahtev za spremembe povezuje življenjski cikel podane zahteve za spremembo PO z življenjskimi cikli nalog, s katerimi se omenjena zahteva za spremembo izvede (gl. sliko 34).

Slika 34: Povezava življenjskega cikla spremembe z življenjskimi cikli nalog



Vir: Taborda, 2001, str. 45

5.3 Predstavitev variant

V nadaljevanju so v kratkem predstavljene izbrane variante, obravnavane v odločitvenem modelu (glej podpoglavje 5.1). Pri predstavitvi variant so poleg lastnih ugotovitev v ustrezni meri upoštevani viri, uporabljeni tudi pri predstavitvi osnovnih modulov upravljanja konfiguracij v prejšnjem podpoglavju, ocene zunanjih svetovalcev ter analize, izvedene v okviru podjetja.

5.3.1 PDM Sherpa in lastni SCM - nadaljevanje razvoja z izvedbo nadgradnje

To varianto sestavljata PDM aplikacija Sherpa PIMS in lokalni SCM sistem, katerega osnovo predstavlja SCCS. Rešitev bi pomenila povsem novo zasnovo SCM/PDM orodja (Pučko, 2002, str. 16). Realizacija bi zahtevala temeljito analizo obstoječega SCM/PDM procesa, vključno z vsemi postopki na operativnem nivoju, pripravo celovitega koncepta izboljšanega procesa in orodja, podrobno definicijo procesa z vsemi postopki ter izvedbo vseh faz razvoja orodja.
Dobra lastnost te rešitve je v velikih možnostih prilaganja obstoječim postopkom, kjer je to smiselno. Uporabniki so že vajeni uporabniških vmesnikov sistema, poleg tega so navajeni na koncepte in terminologijo, ki se uporabljajo v zvezi s sistemom.
Tveganje v zvezi s tem sistemom je predvsem v negotovosti glede možnosti realizacije zadostne funkcionalnosti v sprejemljivem času (dve leti), posebej glede na število razpoložljivih izvajalcev. Med potencialne slabosti tega sistema lahko štejemo predvidevanje, da bo v primeru njegove nadaljnje uporabe težje pridobiti uporabnike za kakršne koli korenitejše spremembe v zvezi z izboljšavami v procesu (Pučko, 2002, str. 16).
Možnost, da v hiši razvit SCM/PDM postane tudi komercialni Iskratelov produkt, je glede na vložke, potrebne za realizacijo, in izjemno visoko strokovno zahtevnost, majhna. Potrebno bi bilo rešiti razmeroma velike probleme v zvezi z možnostmi boljše podpore dela na večih platformah, podporo distribuiranega razvoja, boljšimi možnostmi v zvezi s transparentnim načinom dela na programskih paketih ipd. Za uspešnost nadaljnjega razvoja in uporabe te variante bi bilo zelo zaželjeno sodelovanje zunanjih svetovalcev - specialistov za postopke in sisteme za upravljanje konfiguracij izdelkov in njihove PO.

5.3.2 PDM Teamcenter in SCM ClearCase

To varianto sestavljata sistema EDS Teamcenter in Rational ClearCase z vgrajenim UCM. SCM funkcionalnost pokriva orodje ClearCase, PDM funkcionalnost Teamcenter. ClearCase vključuje upravljanje konfiguracij na ravni programske opreme, vključno s podporo razvojnim delovnim okoljem ter gradnji in izdajanju PO. Teamcenter vključuje tudi upravljanje konfiguracij celotnega proizvoda, materialne opreme, upravljanje sprememb, razvojnega procesa, problemov in dokumentov.
Dobre lastnosti tega sestavljenega sistema so: kakovostna SCM funkcionalnost, lažji prehod za uporabnike zaradi podobnosti aplikacije Teamcenter s Sopranom (Pučko, 2002, str. 16) in razmeroma dobre možnosti za medsebojno povezovanje obeh sistemov (glej razdelek 5.2.4.3). Tveganje se navezuje na relativno visoko ceno nakupa ClearCase-ovih licenc (Pučko, 2002, str. 17) in z omejenimi možnostmi prenosa podpore obstoječega procesa v nov sistem.
Možnosti, da bi ClearCase-ov UCM zadovoljivo podprl obstoječi proces upravljanja konfiguracij programske opreme, so minimalne. Ta podpora bi bila mogoča le ob ustrezni preureditvi tako fizične organizacije programske opreme oziroma programskih paketov proizvodov SI2000 kot tudi samega procesa upravljanja konfiguracij programske opreme.

5.3.3 PDM mySAP PLM in CM Synergy

To varianto sestavljata sistema mySAP PLM (modul sistema mySAP, namenjen upravljanju življenjskega cikla proizvodov) in Telelogic CM Synergy. Upravljanje konfiguracij in sprememb PO pokriva orodje CM Synergy, vse ostale glavne funkcionalnosti v zvezi z upravljanjem proizvodov (upravljanje konfiguracij celotnega produkta, materialne opreme, upravljanje sprememb, razvojnega procesa, problemov in dokumentov) pa modul mySAP PLM.

Dobra lastnost te variante je, da vključuje sistema, ki spadata na svojem področju v najvišji kakovostni razred. V orodju CM Synergy je vgrajena vrsta najboljših izkušenj (best practices) tako s področja upravljanja konfiguracij PO kot glede avtomatizirane podpore inženirskemu procesu razvoja in vzdrževanja PO. Odprta zasnova sistema mySAP daje dobro osnovo za povezovanje obeh orodij v učinkovit sistem.
Tveganja v zvezi s to varianto so povezana z visoko ceno sistema, hkrati pa tudi z razmeroma visokim vložkom v svetovanje. Poleg tega obstaja odprto vprašanje v zvezi z dejansko izrabo vseh razpoložljivih funkcionalnosti v orodjih. Posebej zasnova celovitega koncepta upravljanja sprememb proizvodov in njihove programske opreme in v tem smislu izvedena integracija sistemov sta prepuščeni nam. Pri tem smo omejeni z že implementiranim procesom v dveh različnih orodjih. Od uspešnosti rešitve tega problema je odvisna uspešnost celotne rešitve.

5.3.4 PDM mySAP PLM in SCM ClearCase

Ta varianta se od prejšnje razlikuje v SCM modulu, kjer namesto CM Synergy nastopa Rational ClearCase z dodanim UCM. Razen upravljanja konfiguracij programske opreme so vse ostale funkcionalnosti podprte v SAP (Pučko, 2002, str. 18).
Prednost te variante pred ostalimi je, da vključuje najkakovostnejša orodja, ki jih je v tem razredu produktov možno dobiti na tržišču. Tudi pri tej varianti je pomanjkljivost v ceni nakupa in svetovanja. Prav tako kot pri prejšnji varianti bi bilo tudi pri tej potrebno postaviti koncept celovitega in povezanega upravljanja sprememb, tako za celotne proizvode kot za PO.
Tveganja v zvezi s to varianto so povezana z višino in upravičenostjo naložbe. Zasnova celovitega koncepta in integracija sta zaradi dveh ločenih orodij prepuščeni nam. Od uspešnosti je odvisna uspešnost celotne rešitve. Hkrati nas z že implementiranim procesom v dveh različnih orodjih pri zasnovi precej omejujeta. Dobro pri tem je, da imata oba sistema dokaj odprto zgradbo, zato pri njunem povezovanju pri njunem povezovanju verjetno ne bi naleteli na nepremostljive ovire. Prav tako kot pri drugi varianti bi bilo potrebno ustrezno preurediti tako datotečno ureditev PO opreme kot tudi samega procesa upravljanja konfiguracij PO.

5.3.5 PDM Teamcenter in lastni SCM

Ta rešitev zahteva najprej temeljito analizo obstoječega SCM/PDM procesa, obstoječih sistemov in funkcionalnosti orodja SAP. Sledi zasnova celovitega koncepta izboljšanega procesa in SCM orodja, pri čemer upoštevamo funkcionalnosti orodja SAP, skupaj z v orodje vgrajenim procesom. Pri zasnovi koncepta je potrebno upoštevati prenos funkcionalnosti sedanjega ECC v drugo orodje (npr. v ARS Remedy), pri tem mora biti zagotovljena podpora obvladovanju problemov in upravljanju sprememb. Izvesti bi bilo potrebno obsežno modernizacijo lastnega SCM in povezavo s SAP ter ostalimi sistemi za podporo poslovanju.
Rešitev je zaradi višine naložbe v licence orodja SAP smiselna samo kot dolgoročna. Za uspešnost izvedbe je potrebno sodelovanje zunanjih svetovalcev.

5.4 Rezultati vrednotenja

5.4.1 Ocene variant

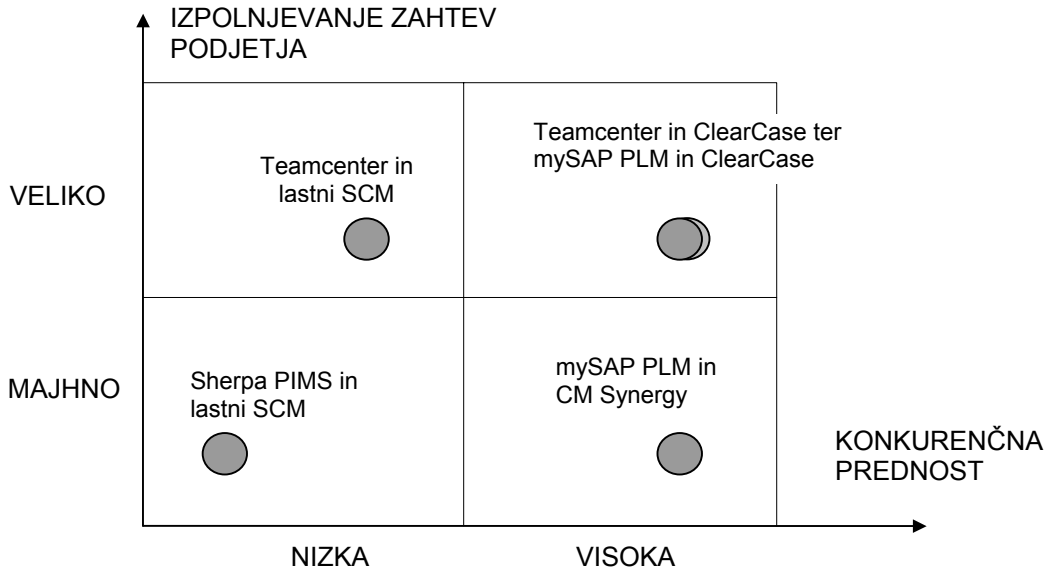
Tabele z rezultati vrednotenja so prikazane v prilogi B. Pri ocenjevanju je bil uporabljen odločitveni model, predstavljen v četrtem poglavju. Modeliranje je bilo izvedeno z ekspertnim

sistemom DEX. Model v DEX-u, ki vključuje drevesa kriterijev, funkcije koristnosti in ocene izbranih variant (glej podpoglavje 5.3), je v celoti nameščen na priloženi zgoščenki.
V zvezi z dobljenimi ocenami (glej sliko 35) lahko ugotovimo naslednje:

- glede izpolnjevanja zahtev podjetja (na voljo so bile ocene “zelo slabo”, “slabo”, “dobro in "zelo dobro") so se dobro odrezali sistemi "Teamcenter in lastni SCM", "mySAP PLM in ClearCase" ter "Teamcenter in ClearCase". V tem pogledu sta bili zelo slabo ocenjeni varianti "mySAP PLM in CM Synergy" ter "Sherpa PIMS in lastni SCM". Vsaka od njiju ima sicer po nekaj dobrih lastnosti (varianta "Sherpa PIMS in lastni SCM" se npr. odlikuje po dobri podpori modularnosti oziroma funkcionalnih modulov in po skladnosti v sistem vgrajenega procesa upravljanja konfiguracij z dejanskim procesom; varianta "mySAP PLM in CM Synergy" ponuja celovite rešitve za upravljanje konfiguracij in dobre možnosti glede integracije v informacijski sistem podjetja, poleg tega ponuja dobre uporabniške vmesnike), vendar pa sta se pokazali kot pomanjkljivi v nekaterih pomembnih kriterijih. Varianta "Sherpa PIMS in lastni SCM" ima precej šibkih točk, med njimi: slabo pokrivanje razvojnih platform; neintuitivne uporabniške vmesnike; slabe možnosti podpore geografsko porazdeljenega razvoja; za aplikacijo Sherpa PIMS ni več zagotovljene podpore (slaba perspektiva). Varianta "mySAP PLM in CM Synergy" pa ima naslednje pomembne pomanjkljivosti: zelo visoka investicija, pri čemer v finančnem pogledu ne moremo računati na znatne ugodnosti; majhna skladnost procesa upravljanja konfiguracij, implementiranega v orodju, z dejanskim procesom, ki teče v podjetju; poleg tega je ta sistem razmeroma kompleksen za uporabo, kar skupaj s prejšnjo ugotovitvijo pomeni, da bi pri prehodu na ta sistem lahko naleteli na resne težave;
- v zvezi s konkurenčno prednostjo orodij (pri tem kriteriju so bile možne ocene “zelo nizka”, “nizka”, “visoka” in “zelo visoka” konkurenčna sposobnost) so bili kot zelo visoko konkurenčni ocenjeni naslednji podporni sistemi: "mySAP PLM in ClearCase", " mySAP PLM in CM Synergy" ter "Teamcenter in ClearCase". Konkurenčna sposobnost sistema "Teamcenter in lastni SCM" je bila ocenjena kot nizka, medtem ko je bil sistem "Sherpa PIMS in lastni SCM" ocenjen kot zelo nizko konkurenčen. Pri tem sistemu je bila iz razumljivih razlogov (Sherpa PIMS je mrtva aplikacija, domači SCM ima zastarelo zasnovo, razvojni viri zanj pa so zelo omejeni) tržna pozicija ocenjena kot slaba, poleg tega ima zelo slabe možnosti za modeliranje procesov oziroma za fleksibilno podporo različnim tipom procesov in življenjskih ciklov. Kljub vsemu ima tudi ta sistem dobre lastnosti, v primerjavi z ostalimi variantami je npr. boljši glede podpore.

Ob upoštevanju obeh glavnih kriterijev lahko ugotovimo, da sta najbolje ocenjeni varianti "Teamcenter in ClearCase" ter "mySAP PLM in ClearCase". V obeh primerih gre za kombinacijo predstavnika vodilnih PDM/PLM sistemov z enim od vodilnih SCM sistemov. V nadaljevanju je podana njuna medsebojna primerjava in primerjava najboljše variante z ostalimi.

Slika 35: Portfeljska matrika z ocenjenimi sistemi orodij za upravljanje konfiguracij in njihove PO



5.4.2 Najboljša varianta; primerjava s preostalimi variantami

Kot navedeno v prejšnjem razdelku, sta bila izmed petih sistemov orodij za upravljanje konfiguracij izdelkov in njihove programske opreme najbolje ocenjena "Teamcenter in ClearCase" ter "mySAP PLM in ClearCase". Glede na to, da sta ta dva sistema dobila enako “skupno” oceno, poskusimo nekoliko od bliže pogledati, ali sta ti dve orodji zares enako dobri.

Če pogledamo ključne kriterije v zvezi z zahtevami podjetja, lahko opazimo, da gre za precej enakovredna sistema, vendar pa obstaja tudi nekaj razlik:

- možnosti za integracijo v celovit informacijski sistem podjetja so pri sistemu "mySAP PLM in ClearCase" ocenjene bolje kot pri "Teamcenter in ClearCase". Razlog je v tem, da prva različica nudi boljše možnosti za povezovanje z obstoječimi sistemi za podporo poslovanja. V podjetju se namreč sistem SAP na nekaterih področjih (finance, logistika...) že uporablja;
- obseg investicije bi bil pri varianti "mySAP PLM in ClearCase" nekoliko višji, hkrati pa ocenjujemo, da bo njena donosnost (upoštevano je petletno obdobje uvajanja in uporabe) tudi nekoliko višja kot pri drugi od omenjenih variant. Večjo donosnost naj bi med drugim dosegli z boljšo izkoriščenostjo povezanega SAP-ovega sistema;
- tudi ocena kompletnosti uporabniških vmesnikov je pri sistemu "mySAP PLM in ClearCase" nekoliko višja kot pri varianti "Teamcenter in ClearCase". Sistem mySAP na tem področju trenutno ponuja nekoliko bolj celovite rešitve kot Teamcenter.

Na podlagi navedenega se lahko zaključi, da je različica "mySAP PLM in ClearCase", kar se tiče zahtev podjetja, za odtenek primernejša od kombinacije "Teamcenter in ClearCase". Do podobnega zaključka lahko pridemo, če primerjamo ti dve orodji po kriterijih v zvezi s konkurenčno sposobnostjo. Pri večini dejavnikih sta si sistema precej enakovredna, kar je povezano s sorodnimi pristopi, ki so bili upoštevani pri snovanju sistemov mySAP in Teamcenter. Funkcionalnosti obeh sistemov, pa tudi poslovna uspešnost sistemov in proizvajalcev, sta bili enako ocenjeni. Pri varianti "mySAP PLM in ClearCase" je nekoliko bolje ocenjena zgradba sistema, predvsem zaradi bolje ocenjene odprtosti sistema (uporaba transparentnih mehanizmov za prenašanje podatkov tako med deli sistema kot med sistemom

in zunanjimi aplikacijami). Na podlagi omenjenih dejstev lahko zaključimo, da je sistem "mySAP PLM in ClearCase" konkurenčno nekoliko sposobnejši kot sistem "Teamcenter in ClearCase".
Predno podamo končno oceno, katero izmed obeh orodij je boljše, si poglejmo še, kako sistema ustrezata najpomembnejšim kriterijem. Na te kriterije in njihovo pomembnost smo že opozorili, nekateri izmed njih so našteti tudi v ciljih. Predvsem gre tu za: celovitost rešitve; funkcionalnosti in zgradbo sistema; integracijo v sistem; višino investicije; težavnost prehoda; zadovoljstvo uporabnikov; skladnost procesa; prilagodljivost sistema; perspektivo sistema in njegovega proizvajalca. Ocene za te kriterije so razvidne iz tabel v prilogi B. Obe različici sta si po teh kriterijih zelo enakovredni. Sistem "mySAP PLM in ClearCase" je boljši pri kriteriju 'integracija v sistem', vendar pa je obseg investicije pri tej varianti višji kot pri različici "Teamcenter in ClearCase". To pomeni dodatno tveganje v zvezi z investicijo v ta sistem. Po drugi strani pa je donosnost investicije v sistem "mySAP PLM in ClearCase" ocenjena višje. Na podlagi omenjenih ugotovitev lahko povzamemo, da je različica "mySAP PLM in ClearCase" v splošnem nekoliko boljša od različice "Teamcenter in ClearCase". V zameno za nekoliko višje tveganje pri njej dobimo višjo donosnost.
Če primerjamo sistem "mySAP PLM in ClearCase" z ostalimi tremi sistemi, lahko opazimo, da:

- v primerjavi s sistemom "Sherpa PIMS in lastni SCM" ponuja boljšo zgradbo in boljše pokrivanje funkcionalnosti upravljanja konfiguracij in sprememb izdelkov. Boljša je tudi podpora projektnemu načinu dela. Ponuja bolj celovite rešitve in boljše možnosti integracije v sistem, tako glede razvojnih platform kot glede povezovanja z drugimi sistemi za podporo poslovanja in z razvojnimi okolji. Edina slabost v primerjavi s sistemom "Sherpa PIMS in lokalni SCM" je v večji težavnosti prehoda na nov sistem. Razlog je v pomanjkljivi skladnosti procesov, vgrajenih v sistem "mySAP PLM in ClearCase", s procesi, kot se trenutno vodijo v podjetju;
- v primerjavi s sistemom "Teamcenter in lastni SCM" ponuja boljše pokrivanje funkcionalnosti upravljanja konfiguracij in sprememb, predvsem na področju upravljanja konfiguracij PO, delovnih okolij za razvoj PO in gradnje PO. Prav tako ponuja boljše možnosti za podporo projektov, v katerih se razvija PO. Zgradba sistema (posebej kar se tiče SCM modula) je naprednejša in bolj odprta. Ponuja boljše možnosti integracije v sistem, tako kar se tiče pokrivanja razvojnih platform kot kar se tiče povezovanja z drugimi sistemi za podporo poslovanja pa tudi z razvojnimi okolji;
- v primerjavi s sistemom "mySAP PLM in CM Synergy" je uporaba sistema nekoliko manj kompleksna. Ob približno enakem enakem obsegu investicije lahko v primeru sistema "mySAP PLM in ClearCase" pričakujemo določene finančne ugodnosti, vezane predvsem na nakup ClearCase-ovih licenc.

Sistem "mySAP PLM in ClearCase" torej od variant "Sherpa PIMS in lastni SCM" ter "Teamcenter in lastni SCM" bistveno odstopa, tako kar se tiče zgradbe in funkcionalnosti kot kar se tiče zahtev podjetja, od variante "mySAP PLM in CM Synergy" pa v pozitivnem smislu odstopa pri nekaterih dejavnikih v zvezi z zahtevami podjetja.
Za zaključek lahko rečemo naslednje: če bi šli v projekt moderniziranja našega sistema orodij za podporo upravljanju konfiguracij izdelkov in njihove PO v tem trenutku (projekt vrednotenja sistemov ravno teče, predviden je tudi pilotski projekt s tem namenom), bi po strokovni plati predlagali dodaten praktični preizkus obeh variant, tako "mySAP PLM in ClearCase" kot "Teamcenter in ClearCase". Prva od obeh ponuja nekoliko boljše rešitve v zvezi z zgradbo in integracijo sistema v naš celotni informacijski sistem, druga pa za nekoliko nižjo ceno ponuja enakovreden oziroma zelo primerljiv nabor funkcionalnosti kot prva od omenjenih rešitev.

6. Kritična analiza modela, metodologije in narave odločanja

6.1 Analiza modela in odločitve

6.1.1 Ustreznost modela in obrazložitev odločitve

Glede na to, da sem v preučevanje področja upravljanja konfiguracij in sprememb izdelkov in njihove vložene programske opreme, pa tudi v ovrednotenje sistemov orodij za avtomatizacijo omenjenega procesa v preteklem letu dni in pol vložil kar nekaj časa, sem v veliki meri prepričan, da so kriteriji ustrezno izbrani in strukturirani, uporabljene funkcije koristnosti pa ustrezno definirane. To seveda ne pomeni, da se modela ne bi dalo na veljaven način postaviti tudi kako drugače. Model in njegova uporaba po mojem mnenju predstavljata kakovostno osnovo za odločanje o primernem sistemu orodij za upravljanje konfiguracij izdelkov SI2000 in njihove najpomembnejše komponente, programske opreme.

Izbrana metoda portfelja nam, po izkušnjah sodeč, daje pregledne, intuitivne rezultate, na podlagi katerih lahko uporabimo v naprej izbrane strategije. Za oba ključna oziroma končna kriterija (zahteve podjetja, konkurenčna prednost) sem izbral štiristopenjski ocenjevalni lestvici. Razloga za to sta vsaj dva. Prvi je v tem, da se s sodim številom ocen izognemo srednjim ocenam, ki bi nam pri izbrani metodi portfelja lahko delale preglavice. Drugi je v tem, da bi z ozirom na razpoložljive informacije in praktične izkušnje o sistemih v tem trenutku težko izdelali bolj natančne ocene za tako kompleksen in obsežen problem. Z 'drobljenjem' lestvice bi se povečala nevarnost, da pri pri gradnji modela in njegovi uporabi pridemo do nekonsistentnih rezultatov.

Lahko rečemo, da so bili rezultati, dobljeni s pomočjo modeliranja z ekspertnim sistemom DEX, v okviru mojih pričakovanj. V zvezi z vrednostjo atributov je potrebno omeniti, da so bile le deloma pridobljene na podlagi praktičnih preizkusov v resničnem okolju. V večini primerih so bile ocene dobljene na podlagi posrednih informacij (dosegljivi pisni in ustni viri). Kljub temu gre v večini primerov za razmeroma natančne ocene. Nekateri kriteriji so z vidika natančnosti nekoliko problematični. Tak je npr. tudi kriterij v zvezi z donosnostjo investicije oziroma vrednost kazalnika ROI. V okviru podjetja je v tem trenutku razmeroma težko dobiti vse potrebne informacije, potrebne za izračun neposrednih koristi, ki bi jih imeli z dano investicijo v informacijsko tehnologijo. Pri natančnem izračunu tovrstnih koristi bi bilo potrebno izhajati iz dobro definiranega in ovrednotenega razvojnega procesa. Le tako bi lahko vrednostno primerjali ta proces z drugimi različicami razvojnega procesa, ne glede na njegovo podprtost z računalniškimi orodji. Ker že prvi od navedenih pogojev (t.j. dobro določen proces) ni izpolnjen, je ocena neposrednih koristi investicije lahko le približna. Še neprimerno bolj zapleten je izračun posrednih koristi investicije (npr. kakovost izdelkov, višje zaupanje kupcev, hitrejša odzivnost na tržne priložnosti, višja konkurenčnost podjetja ipd.). Tovrstne prednosti je težko finančno ovrednotiti, za njihovo natančnejšo oceno pa bi morali imeti na razpolago vrsto informacij in parametrov, s katerimi pa podjetje v tem trenutku (še) ne razpolaga. Iz omenjenega sledi, da je izračun ROI lahko le približna ocena. Izdelana je bila na podlagi različnih opravljenih izračunov, ki so zajeti v interni dokumentaciji podjetja. Na srečo ta dejavnik v modelu nima največje pomembnosti in model nanj ni zelo občutljiv. Odločitveni model je namreč usmerjen v tehnični, informacijsko-upravljalski vidik. Kljub vsemu pa ima tudi izračun kazalnika ROI določeno

pomembnost. Pomeni pomemben pokazatelj v zvezi z investicijo za vodstvo. Velik je njegov pomen pri finančnem ovrednotenju tistih dejavnikov, povezanih z investicijo, ki nastopajo pri izračunu tega kazalnika. Hkrati je izračun kazalnika ROI pomemben z vidika dolgoročnega upravljanja informacijske tehnologije, saj predvidevamo, da bomo na podlagi izkušenj in pridobivanja dodatnega ekonomskega znanja iz projekta v projekt bolj sposobni oceniti ekonomsko upravičenost investicij.

6.1.2 Prednosti in pomanjkljivosti variant

V nadaljevanju so naštete prednosti in pomanjkljivosti posameznih variant oziroma sistemov, ocenjevanih s predstavljenim ocenjevalnim modelom:

- sistem "Sherpa PIMS in lastni SCM" ima naslednje dobre lastnosti v primerjavi z ostalimi sistemi: dobro podporo pri vpeljavi s strani domače ekipe sistemskih inženirjev; dobro podporo obstoječi metodologiji v zvezi s spreminjanjem programskih paketov, modulov; veliko skladnost procesa, vgrajenega v orodje, z obstoječim procesom. Ima pa naslednje šibke točke: slabo pokrivanje razvojnih platform za programsko opremo (zadovoljivo pokriva le HP-UX); slabe uporabniške vmesnike (vmesniki PDM sistema ne dovolj intuitivni, v SCM sistemu vmesniki, slabo podprti z grafiko); na sistem dolgoročno ne moremo računati, saj je Sherpa PIMS mrtva aplikacija; slabo podprte možnosti za podporo različnim tipom procesov in življenjskih ciklov gradnikov in drugih objektov;
- sistem "Teamcenter in ClearCase" se odlikuje po naslednjih prednostih: je uveljavljen sistem uveljavljenih proizvajalcev; dobro podpira tako upravljanje konfiguracij izdelkov kot upravljanje konfiguracij PO; dobro podpira delo na projektih; ima trdoživo zgradbo, ki omogoča dobro podporo razvoju izdelkov in njihove PO na porazdeljenih geografskih lokacijah; dobro podpira razvojne platforme; pri nakupu ClearCase-ovih licenc lahko računamo na finančne ugodnosti. Slabost tega sistema je, da bi z njim težko zadovoljivo podprli obstoječi način izvajanja procesa upravljanja konfiguracij in sprememb izdelkov in še posebej njihove PO. V ta namen bi bilo potrebno izvesti nekaj sprememb tako v sistemu kot v omenjenem procesu;
- sistem "mySAP PLM in CM Synergy" se odlikuje po dobrih možnostih povezovanja z obstoječimi sistemi za podporo poslovanju v podjetju, po visoko avtomatizirani podpori procesu upravjanja konfiguracij izdelkov in njihove PO; po podpori različnih tipov procesov in življenjskih ciklov; po napredno zasnovani zgradbi sistema in po komponentno in v naloge usmerjenem upravljanju konfiguracij PO. Njegovi pomembni pomanjkljivosti sta visoka cena in razmeroma visoka kompleksnost uporabe SCM modula;
- varianta "mySAP PLM in ClearCase" se odlikuje po dobrih možnostih povezovanja z obstoječim informacijskim sistemom podjetja; po dobrih možnostih uporabe različnih tipov uporabniških vmesnikov; po dobri podpori velikih razvojnih projektov v geografsko porazdeljenem poslovnem sistemu; po napredno zasnovani zgradbi obeh modulov sistema in po zelo visoki tržni uveljavljenosti ter dobri perspektivi njunih proizvajalcev. Glavni slabosti tega sistema sta njegova visoka cena in majhne možnosti za podporo obstoječih procesov v podjetju, brez da bi le-te v večji meri prilagodili sistemu in vanj vgrajenim zmožnostim za podporo oziroma avtomatizacijo procesa;
- dobra lastnost variante "Teamcenter in lastni SCM" je, da za razmeroma majhen vložek dobimo dobre možnosti za podporo razvoja izdelkov v geografsko porazdeljenem porazdeljenem poslovnem okolju, pri čemer ohranimo vse prednosti obstoječega SCM sistema (npr. podpora načinu dela s programskimi paketi in skladnost z obstoječim

procesom). Pri tem ocenjujemo, da bi s tem sistemom izboljšali raven podpore procesu upravljanja zahtev za proizvode. Največja slabost teda sistema je, da bi podedoval nekatere ključne slabosti SCM sistema, npr. slabo podporo razvojnih platform, slabosti v zgradbi ipd.

Splošna ugotovitev, da nastopajoče različice oziroma ocenjevani podporni sistemi ne ustrezajo najbolje zahtevam našega razvojnega sistema, sama po sebi ne pomeni, da imajo ti podporni sistemi kakšne pomembne pomanjkljivosti. Omenjeno dejstvo lahko izhaja iz tega, da je tako naš način vodenja produktne linije proizvodov kot tudi proces razvoja in vzdrževanja izdelkov SI2000 razmeroma edinstven in da ne uporabljamo povsem standardnih, splošnih prijemov pri upravljanju tega procesa. Prav tako pa na zapletenost situacije v dodatni meri vpliva dejstvo, da razmere niso idealne na drugih procesnih področji, posebej na tistih, ki so z upravljanjem konfiguracij najbolj soodvisni: upravljanje zahtev za proizvode, upravljanje produktne linije, upravljanje z zgradbo proizvodov, vodenje razvojnih projektov ipd.

Omenjene ugotovitve seveda po drugi strani ne pomenijo, da je nekaj 'narobe' z razvojnim ali poslovnim okoljem. To je kot vsa poslovna okolja edinstveno, proizvaja edinstvene izdelke za svoje naročnike, ki dobro poznajo in znajo izraziti svoje zahteve v zvezi s proizvodi. Podjetje je v preteklosti znalo dobro izkoristiti rapoložljive potenciale in vire, vključno s sistemi za podporo življenjskega cikla izdelkov SI2000. To se je ne nazadnje potrdilo z uspešnim izvajanjem poslovnih strategij in trženjem izdelkov SI2000 na mnogih svetovnih tržiščih. Ker pa kupci oziroma uporabniki v splošnem postajajo vse zahtevnejši in pričakujejo vse večjo kakovost tako storitev kot izdelkov, se kaže vse večja potreba po uporabi standardnih, preizkušenih in potrjenih načinov dela oziroma poslovanja. Mednje prav gotovo spada uporaba najboljših izkušenj ("best practices") na različnih procesnih področjih, od upravljanja zahtev, sprememb in konfiguracij na izdelkih do planiranja in upravljanja projektov, tveganja in kakovosti. Na teh področjih poslovanja se za podjetje kažejo številne priložnosti za izboljšave. Te bodo za podjetje na začetku sicer predstavljala določene stroške, v dolgoročnem smislu pa investicijo v še učinkovitejši, kakovostnejši in cenejši razvoj izdelkov/storitev, s tem pa tudi v še višjo konkurenčno sposobnost podjetja.

6.1.3 Bistvene razlike med variantami

Med variantami so se pri nekaterih kriterijih pokazale razmeroma velike razlike:

- *višina investicije.* Investicija v sistema, v katerih nastopa lastni SCM modul, je v primerjavi z morebitnimi investicijami v druge različice sistemov nižja. En glavnih vzrokov za to je v tem, ker stroški, povezani z lastnim SCM sistemom, ne vključujejo nakupa dodatnih licenc za uporabo tega sistema in so v glavnem vezani na stroške vzdrževanja, izvajanja nadgradenj in uporabe. Najvišji sta investiciji v sistema "mySAP PLM in ClearCase" ter "mySAP PLM in CM Synergy". V obeh primerih gre za visoke stroške uvajanja sistemov - nakup licenc, svetovanje, šolanja uporabnikov...;
- *donosnosti investicij* so za različne variante zelo različno ocenjene. V primeru različice "Sherpa PIMS in lastni SCM" je ocena najnižja, povezana je z navedenimi omejitvami sistema, ki bodo v prihodnje vse pomembnejše. Pri tem je še posebej veliko tveganje povezano z uporabo PDM modula, saj kot že omenjeno zanj ni več zagotovljeno vzdrževanje s strani proizvajalca. Najvišje je ocenjena donosnost investicij pri sistemih "mySAP PLM in ClearCase" in "mySAP PLM in CM Synergy". En od vzrokov za njuno najvišjo oceno je v tem, da gre v obeh primerih za povezavo vrhunskega PLM/PDM sistema z vrhunskim SCM sistemom. Drug pomemben vzrok je v tem, da bi s tema sistemoma lahko izkoristili veliko prednosti povezanega sistema za podporo poslovanja

podjetja, saj se SAP-ovi moduli že uporabljajo na področju logistike, financ in drugih področjih. Poleg tega so pri tem upoštevane mnoge posredne koristi uporabe teh sistemov, ki pa jih je težko natančno ovrednotiti: nižje število napak na produktih, višja konsistentnost in kakovost izdelkov, večje zaupanje kupcev v naše izdelke, hitrejše odzivanje podjetja na tržne priložnosti, višja konkurenčna sposobnost podjetja ipd.;

- *uporabnost rešitve.* Ocenjujemo, da bomo sistem "Sherpa PIMS in lastni SCM" prisiljeni v najkasneje dveh letih zamenjati z drugim sistemom. Glavni vzrok za to je v tem, da je Sherpa PIMS 'mrtva aplikacija'. Sistem "Teamcenter in lastni SCM" lahko predvidoma preživi obdobje od dveh do petih let, nakar bo potrebno narediti ali zelo velike posege v SCM modul (npr. prilagoditi ga za boljšo podporo geografsko porazdeljenega načina razvoja PO, temu prilagoditi njegovo zgradbo, omogočiti bolj transparenten način dela s PO, izboljšati nadzor nad delovnimi okolji uporabnikov, povečati učinkovitost gradnje PO itd.) ali pa kupiti komercialni SCM sistem. Ostale tri variante, v katerih nastopajo sami zelo uveljavljeni PDM in SCM moduli, so ocenjene kot dolgoročne;
- *skladnost procesa.* Pri varianti "Sherpa PIMS in lastni SCM" je ta skladnost zelo velika iz preprostega razloga, ker se omenjeni sistem že nekaj časa uporablja za avtomatizirano podporo našemu procesu upravljanja konfiguracij in sprememb izdelkov SI2000, vključno z njihovo PO. Za varianto "Teamcenter in lastni SCM" je ta kriterij ocenjen kot delno skladen. Proces je tu razmeroma skladen vsaj na ravni upravljanja konfiguracij PO. Procesi, realizirani v ostalih CM sistemih, so le v majhni meri skladni z našim procesom. V teh primerih gre običajno za skladnost tistih pravil dobrih praks na področju upravljanja konfiguracij in sprememb izdelkov in njihove PO, realiziranih v teh sistemih, ki se uporabljajo v obstoječem procesu in so že podprti s trenutnim CM sistemom. Za ustrezno avtomatizirano podporo procesu bi bilo potrebno izvesti tako določene posege v te variante oziroma sisteme kot tudi v naš proces in sistem upravljanja konfiguracij izdelkov in njihove PO. Ker je ta proces še zlasti na področju PO zelo prepleten z razvojnim procesom, bi bil ta poseg za podjetje razmeroma zahteven, dolgotrajen in ne nazadnje drag;
- glede *perspektive proizvajalca* lokalnega CM sistema lahko rečemo, da je ta v primeru "Sherpa PIMS in lastni SCM" slaba - pri tem glavno oviro predstavlja zastarel PDM sistem Sherpa. Proizvajalec tega sistema je bil prevzet s strani drugega podjetja, sistem ne bo več podprt, možnosti prilagoditev s strani domače skupine inženirjev so omejene. Tudi perspektiva 'proizvajalcev' lokalnega SCM sistema ni visoka, predvsem iz razloga, ker gre za majhno skupino inženirjev, ki ne more vlagati dovolj sredstev v razvoj ali vsaj sledenje novih tehnologij. To vpliva na kvečjemu 'povprečno' oceno perspektivnosti 'proizvajalca' sistema 'Teamcenter in lastni SCM'. V vseh drugih primerih gre za perspektivne proizvajalce, z velikimi vložki v nove tehnologije;
- *podpora pri vpeljavi* bi bila razumljivo najboljša v primeru nadaljnje uporabe sistema "Sherpa PIMS in lastni SCM", saj bi bilo ustrezno znanje v osnovi že na razpolago v podjetju. V vseh drugih primerih je podpora ob vpeljavi ocenjena kot povprečna - predvsem iz razloga, ker bi bili v vsakem primeru tuji svetovalci na razpolago vedno le v omejenem obsegu, nastopale pa bi tudi druge ovire, izhajajoče npr. iz 'nehomogenosti' uvajanega CM sistema (njegove PDM in SCM komponente);
- obe različici, v katerih nastopa lokalni SCM, dobro oziroma v primerjavi z vsemi drugimi različicami najbolje podpirata posebne zahteve v zvezi z našim načinom dela z moduli in komponentami (dejavnik 'modularnost') proizvodov SI2000 - predvsem na področju PO, kjer so zahteve najostrejše, saj gre za najkompleksnejšo, 'najmehkejšo' in s tem najtežje

obvladljivo komponento. Ostale različice imajo v ta namen nekaj privzetih možnosti in avtomatizmov, ki pa le v omejeni meri izpolnjujejo omenjene posebne zahteve;

- *podpora distribuiranemu načinu razvoja* je pri lastnem sistemu ocenjena kot slaba, predvsem zaradi omejitev lokalnega SCM sistema z zastarelo zgradbo, temelječo na SCCS. Podpora je pri različici 'Teamcenter in lastni SCM sistem' ocenjena kot pogojno sprejemljiva. Razlog za to je v tem, ker Teamcenter sam po sebi ponuja določene možnosti pri izrabi geografsko razporejenih odlagališč SCM podatkov - lahko jih registrira in izvaja določene operacije v zvezi z njimi, ne da bi jih fizično vseboval. Tudi ostale tri nastopajoče različice so v zvezi s tem dejavnikom ocenjene s 'pogojno sprejemljivo', saj pri vseh nastopajo določene omejitve. V primeru sistema ClearCase je za izrabo te storitve potrebno dokupiti poseben modul MultiSite;
- *prilagodljivost* je največja pri lokalnem sistemu, pri katerem obvladujemo podatkovne strukture in funkcije sistema, za SCM modul pa tudi izvorno kodo, v kateri je sistem sprogramiran. Poleg tega lokalni SCM sistem temelji na podatkovni bazi Informix, katere strukturo je precej lahko spreminjati in prilagajati novim zahtevam. Tudi ostali CM sistemi so delno prilagodljivi, pri čemer je možno uporabljati različne mehanizme, vgrajene v te sisteme (npr. preko opozorilnikov in sprožilcev). Za komercialne sisteme je značilno, da so njihova podatkovna skladišča precej zaprte narave in da njihove strukture ni mogoče spreminjati na povsem enostaven način. S tem proizvajalci poskušajo doseči lažje nadgrajevanje starih verzij teh podpornih sistemov z novimi. Primer: večje kot bi bile spremembe na strukturi odlagališča podatkov in metapodatkov, ki predstavlja enega temeljev kakršnega koli CM sistema, večji bi bili dodatni napori oziroma vložki, s katerimi bi izvedli prenos podatkov iz starega odlagališča v odlagališče, nad katerim bi delovala nova verzija podpornega sistema.

6.1.4 Občutljivost odločitve

Lahko rečemo, da je odločitev v nekaterih pogledih občutljiva. Odgovor na vprašanje, pri katerih elementarnih in sestavljenih kriterijih se ta občutljivost kaže, nam poda t.im. "kaj-če analiza". Primer: Dejstvo, da ni ugodnosti pri nakupu dragega sistema "mySAP PLM in CM Synergy", za to varianto pomeni znatno nižjo oceno v zvezi z izpolnjevanjem zahtev podjetja. Če bi imeli pri nakupu licenc (licence za CM Synergy in dodatne licence za SAP) za ta sistem dovolj ugodnosti, bi bil ta sistem blizu najbolje ocenjenima sistemoma. Ugodnost investicije je torej v tem primeru hkrati nekakšen izločilni dejavnik.

Še ena aktualna hipoteza: V kolikor bi pri sistemu "Sherpa PIMS in lastni SCM" posodobili zgradbo lokalnega SCM sistema, bi po postavitvi ustrezne informacijske infrastrukture (npr. geografsko porazdeljenih odlagališčih SCM podatkov in metapodatkov) in ustreznih funkcij (npr. replikaciji podatkovnih baz omenjenih odlagališč podatkov in metapodatkov) lahko dejavnik v zvezi s podporo geografsko porazdeljenega razvoja ocenili s sprejemljivo ali celo dobro oceno. V tem primeru bi bilo smiselno s pomočjo razvojnega orodja Platinum UIM/X, ki omogoča izdelavo zelo kakovostnih grafičnih vmesnikov, tako za HP-UX kot za okolje Windows, podpreti odjemalce na obeh omenjenih razvojnih platformah. S tem bi v okviru omenjenega sistema v doglednem času lahko dobili zadovoljivo pokritje obeh glavnih razvojnih platform, tako HP-UX kot Windows. Hkrati bi se izboljšala tudi kakovost in intuitivnost samih uporabniških vmesnikov. Pričakovati je, da bi bili strežniki za SCM modul na sistemu z operacijskim sistemom HP-UX, odjemalci pa tako na HP-UX kot na Windows. Tudi dograditev sistema s podporo za Web odjemalce, vsaj za nekatere ključne operacije SCM sistema, v tem primeru ne bi bila neizvedljiva naloga. Ob omenjenih predpostavkah bi se ob sprejemljivem načinu realizacije

izboljšav skupna ocena v zvezi z izpolnjevanjem zahtev podjetja izboljšala za eno oceno, z 'zelo slabo' na 'slabo'. V sistemu bi namreč še vedno ostal nerešen problem z 'mrtvo' aplikacijo Sherpa PIMS. Prehod na novo zgradbo pa bi lahko za seboj potegnil tudi določene dodatne probleme, npr. v zvezi z varnostjo uporabe podatkov omenjenega geografsko porazdeljenega podpornega sistema. Vendar ti problemi po naši oceni ne bi bili nerešljivi, saj so s tem v zvezi v podjetju na razpolago bogate izkušnje.
V splošnem lahko ugotovimo: če spreminjamo vrednosti osnovnih kriterijev, to lahko v precejšnji meri vpliva na rezultate, ki nam jih daje model oziroma funkcija koristnosti. Ta trditev velja še posebej, če spreminjamo vrednosti atributov, ki se nanašajo na najpomembnejše kriterije (funkcionalnost sistema, zgradba sistema, zadovoljstvo uporabnikov ipd.). Primer: Če pri kateri koli varianti znižamo oceno funkcionalnosti sistema na najnižjo možno vrednost, ocena njene konkurenčne sposobnosti pade na najnižjo vrednost. Podobno velja za vse druge pomembne kriterijev
Po drugi strani je funkcija koristnosti na nekatere (nekoliko manj pomembne) kriterije občutljiva le pod določenimi pogoji. Že enkrat omenjen primer, tokrat z drugega zornega kota: višji kot je obseg investicije, pomembneje je, ali obstajajo v zvezi z njo kakšne finančne ugodnosti (npr. časovna razporeditev plačila). Pri zelo dragih sistemih je ta dejavnik zelo pomemben (kar se je pokazalo kot usodno za varianto 'mySAP PLM in CM Synergy'), medtem ko pri nižjih investicijah ta kriterij nima tako pomembne vloge.
V splošnem velja pravilo, da s poslabšanjem ocene pomembnega kriterija lažje zbijemo skupno oceno, kot pa jo s povečanjem ocene povišamo. Za slabo skupno oceno je v nekaterih primerih dovolj (npr. kriterij 'tehnologija'), da je slabo ocenjen en izmed podkriterijev, ki določajo kriterij. Za dobro oceno pa je navadno pogoj, da imajo dobro ali vsaj zadovoljivo oceno vsi faktorji, ki določajo nek kriterij.

6.2 Izkušnje z metodologijo DEX in uporabo orodja DEXi

Z uporabljeno metodologijo, ki je vključevala izgradnjo odločitvenega modela in izvedbo ovrednotenja sistemov s pomočjo sistema DEXi, sem imel zelo pozitivne izkušnje, saj gre za transparentno in razmeroma preprosto, a kljub temu učinkovito in v logičnem ter vsebinskem smislu zelo dobro zastavljeno in konsistentno metodo.
Že sama uporaba metodologije v kakršnem koli projektu uvajanja informacijskih tehnologij nudi celovito, sistematično in povsem transparentno izhodišče za komunikacijo glede problemskega stanja in predmeta odločanja, odločitvenih kriterijev, njihove teže oziroma pomembnosti, izbire in pomembnosti variant ipd.
Reševanja problema sem se lotil po naslednjih glavnih korakih:

- ureditev in preučitev vseh informacij v zvezi s problematiko, ki sem jih imel na voljo;
- zbiranje, evidentiranje kriterijev, ki vplivajo na odločitev;
- izbira smiselnih kriterijev,
- hierarhično strukturiranje kriterijev, z izdelavo odločitvenega drevesa;
- določitev domen za posamezne kriterije;
- formiranje funkcije koristnosti, v katero sem vgradil tudi pomembnost posameznih kriterijev oziroma njihov vpliv na končno oceno;
- določitev seznama aktualnih variant (orodij ali sistemov orodij za upravljanje konfiguracij izdelkov);
- opis variant na podlagi predhodno zbranih informacij;
- ovrednotenje variant;

- interpretacija rezultatov vrednotenja različic;
- preučitev, analiza tako procesa odločanja kot metodologije, uporabljenega modela in dobljenih rezultatov.

Pri tem je potrebno omeniti, da izvajanje naštetih korakov ni teklo linearno, temveč iterativno. Večkrat se je zgodilo, da sem se vrnil na prejšnji korak. Običajna primera:

- pri formiranju funkcije koristnosti sem opazil, da bi bilo mogoče v odločitveno drevo mogoče vnesti nekaj izboljšav, zato sem se vrnil v korak izdelave odločitvenega drevesa;
- pri ocenjevanju variant se je na nekaterih mestih pokazalo, da domene nekaterih kriterijev niso bile izbrane najbolj posrečeno, zato jih je bilo potrebno popraviti.

Orodje DEXi podpira metodologijo na zelo intuitiven in udoben način, pri čemer ima razmeroma bogate možnosti predstavljanja rezultatov modeliranja in ocenjevanja. To omogoča, da iterativno izvajanje korakov v postopku ocenjevanja poteka brez kakršnih koli težav.

Pričakujem, da bosta tako metodologija kot orodje DEX (oziroma DEXi) v okviru podjetja odigrala ustrezno vlogo v nadaljnjih tovrstnih projektih.

6.3 Analiza narave odločanja pri izbiri sistemov za podporo upravljanju konfiguracij izdelkov

V zvezi z analizo narave procesa odločanja sem na podlagi omenjenega, pa tudi prejšnih sorodnih projektov v podjetju, prišel do naslednjih ugotovitev:

- dejavniki, povezani z višino investicije oziroma s ceno sistemov, se zelo pogosto spreminjajo. Zanje velja, da niso transparentni, temveč so odvisni od različnih poslovnih dejavnikov, med drugim lahko tudi od neposrednih pogajanj med ponudnikom in potencialnim kupcem sistema. Ker so ti dejavniki zelo zanimivi in pomembni za vodstvo, to med drugim pomeni, da ta netehnični, poslovni vidik v veliki meri lahko vpliva tudi na sprejem končne odločitve v zvezi z investicijo v uvajanje podpornega sistema;
- višje vodstvo na ravni poslovnih enot podjetja s svojimi načelnimi zahtevami tako v zvezi s procesom odločanja kot v zvezi z odločitvenim modelom (primer: (ne)aktualnost variant, kriteriji odločanja...) običajno postavlja znane robne parametre izvajanju odločitve. Zahteve vodstva, vezane na odločitveni proces, so običajno transparentne in izhajajo iz narave oziroma logike poslovanja. V zvezi s kriteriji vodstvo daje razmeroma velik pomen celovitosti rešitve z ozirom na strateško raven upravljanja z izdelki, poleg tega pa npr. tudi kriterijem finančne narave. Tako imata iz povsem razumljivih razlogov razmeroma velik pomen ocena kazalnika ROI in višina investicije;
- podjetja, ki razvijajo sisteme za področje upravljanja konfiguracij, v zadnjem času namesto z izgotovljenimi izdelki vse raje operirajo s konceptualnimi rešitvami (primer Teamcentra). V svojih "belih knjigah" objavljajo lastnosti svojih izdelkov, ki pa v realnosti še ne obstajajo, temveč so le zasnovani. Šele ko naletijo na dovolj širok krog (kritično maso) zanimanja pri kupcih ali dovolj močnega kupca, začno svojo zasnovo uresničevati v konkretnem poslovnem okolju. To hkrati pomeni, da lahko na vzpostavljeno osnovno infrastrukturo sistema računamo ne prej kot v enem do dveh mesecih, na končno rešitev, z izvedenimi prilagoditvami razvojnemu in širšemu poslovnemu okolju, pa v pol do enega in pol leta. Tak pristop omenjena podjetja uporabljajo vsaj iz dveh razlogov: po eni strani zmanjšajo tveganje (ne tvegajo z vlaganjem virov v razvoj potencialno neuspešnih projektov oziroma izgotovljenih izdelkov), po drugi strani pa se lahko v fazi

seznanjanja z zahtevami naročnikov bolje in podrobneje seznanijo z zahtevami le-teh, jih podrobno preanalizirajo, ocenijo poslovne učinke in jih konec koncev tudi ustrezneje in učinkoviteje realizirajo. Tudi s tem po eni strani kot rečeno zmanjšujejo tveganje (ob predpostavki, da se kupcu ne mudi z uvedbo tovrstnih rešitev), po drugi strani pa ga v določeni meri povečajo, saj se kupec medtem lahko odloči za drugega ponudnika.

7. Zaključek in predlogi za nadaljnje delo

Pričakujem, da bo to magistrsko delo v veliki meri pripomoglo h kvalitetnejši odločitvi v zvezi z izbiro sistema za upravljanje konfiguracij izdelkov SI2000 v podjetju Iskratel, d.o.o., Kranj, prav tako pa tudi k boljšemu razumevanju pomena upravljanja konfiguracij, tako v omenjenem telekomunikacijskem podjetju kot tudi v drugih podjetjih, ki izdelujejo proizvode z vloženimi računalniškimi sistemi. Pri tem je v konceptualnem smislu pomembno ločiti dve ravni: prvo predstavlja proces upravljanja konfiguracij izdelkov in drugo podpornimi sistemi, ki ta proces avtomatizirajo. Pri tem sta tako proces upravljanja konfiguracij kot njegov podporni sistem, vsak na svoji ravni, v veliki meri povezana z drugimi poslovnimi procesi in sistemi za podporo poslovanja v danem podjetju.

S pomočjo različnih izvedenih raziskav je bilo zbranih precej ključnih informacij o obstoječih sistemih in procesih v omenjenem telekomunikacijskem podjetju, ki se uporabljajo v toku življenjskega cikla izdelkov SI2000. Med drugim je bil s tem v podjetju prvič razmeroma dobro identificiran proces upravljanja konfiguracij za omenjene telekomunikacijske izdelke.

Magistrsko delo oziroma postavljeni odločitveni model omogoča nov, razširjen in poglobljen pogled na obravnavano problematiko, in sicer predvsem v zvezi z naslednjim:

- omogoča pregled nad vlogo in pomenom, ki ga ima disciplina upravljanje konfiguracij izdelkov v sodobnih industrijskih podjetjih in posebej v podjetjih, ki izdelujejo proizvode z vloženimi računalniškimi sistemi. Tovrstni proizvodi visoke tehnologije so tipični za telekomunikacijsko, avtomobilsko, letalsko, vojaško pa tudi druge vrste industrij;
- v zgrajenem in uporabljenem modelu so določeni glavni dejavniki in njihov vpliv na ovrednotenje sistemov za upravljanje konfiguracij izdelkov v prej omenjenem podjetju. Z modelom smo s tem dobili transparenten pregled nad pomebnostjo posameznih dejavnikov, ki vplivajo na izbiro najustreznejšega sistema za upravljanje konfiguracij izdelkov SI2000;
- ob ovrednotenju CM sistemov je bil prvič do zdaj izveden poskus ovrednotenja avtomatizirane podpore procesa upravljanja konfiguracij izdelkov SI2000 tudi iz finančnega vidika;
- pod drobnogled je bil vzet sam odločitveni proces. S pridobljenimi izkušnjami se bomo v podjetju lahko še bolj sistematično lotili prihodnjih investicij v sisteme za podporo poslovanju.

Določena pomanjkljivost izvedbe ovrednotenja je, da v razmeroma veliki meri temelji na teoretičnih podatkih in le deloma na praktičnem preizkušanju. Kot omenjeno že v uvodnem poglavju, bi bili preizkusi v večjih projektih ali celo na razvojnih ali vzdrževalnih projektih zelo dragi.

Pričakujem, da bomo tako metodologijo kot podporni sistem DEX uporabili v nadaljnjih tovrstnih projektih. Z njuno uporabo lahko dodatno zvišamo kakovostno raven odločanja pri najkompleksnejših odločitvenih problemih. Tu sta še posebej pomembni sistematičnost in transparentnost odločanja, kar uporabljena metodologija in omenjeni ekspertni sistem, z

možnostmi izbire aktualnih različic in ustreznih kriterijev, njihovega preglednega strukturiranja in določitvijo njihovega vpliva preko funkcije koristnosti, vsekakor omogočata.

V zvezi z nadaljnjimi izboljšavami v okviru obravnavanega področja predlagam naslednjo raziskavo: poiskati optimalno razgradnjo proizvodov SI2000 na njihove komponente, z namenom še učinkovitejšega upravljanja konfiguracij in sprememb celotne družine izdelkov. V zvezi s tem poiskati načine, kako za nove, porajajoče se proizvode družine SI2000 pouporabiti obstoječe in ustvariti morebitne nove komponente. Poiskati načine, kako na najoptimalnejši način organizirati gradnike v teh komponentah, vključno z ureditvijo verzij v okviru posameznega gradnika.

Ta raziskava bi pomagali dodatno dvigniti učinkovitost upravljanja konfiguracij izdelkov SI2000, s tem pa tudi izboljšati možnosti za učinkovito upravljanje njihovega celotnega življenjskega cikla. To bi za podjetje vsekakor pomenilo dodatno povečanje njegove konkurenčne sposobnosti.

Literatura

- Aberdeen Group: Strategy Profile. Boston, Massachusetts, 2001. 10 str.
- Angstadt Bruce: SCM: More than Support and Control. CrossTalk, The Journal of Defense Software Engineering, marec 2000, str. 26-27.
- Asklund Ulf, Magnusson Boris, Persson Annita: The Unified Extensional Versioning Model. SCM-9 Symposium, Toulouse, France, 1999 : proceedings. Berlin : Springer, 1999, str. 100-122.
- Berlack H. Ronald: Software Configuration Management. New York : John Wiley & Sons, 1992. 330 str.
- Bohanec Marko, Rajkovič Vladislav: Multi-Attribute Decision Modeling: Industrial Applications of DEX. Informatica, Ljubljana, 23 (1999), str. 487 - 491.
- Bohanec Marko: Introduction to DEX, An Expert System Shell for Multi-Attribute Decision Making. Univerza v Ljubljani, Institut “Jožef Stefan”, 1991. 12 str.
- Bohanec Marko, Rajkovič Vladislav: Večparametrski odločitveni modeli. Institut “Jožef Stefan” [URL: http://www-ai.ijs.si/MarkoBohanec/org95/index.html], 1995.
- Booch Grady: Unifying Enterprise Development Teams with the UML. Rational Software, Cupertino, CA [URL: http://www.rational.com/media/whitepapers/TP188.pdf], 2000. 7 str.
- Bounds Nadine M., Dart Susan A.: Configuration Management (CM) Plans: The Beginning to Your CM Solution. Pittsburgh, Pennsylvania, Carnegie Mellon University, Software Engineering Institute, 1993. 48 str.
- Burrows Clive: Configuration Management, Coming of Age in the Year 2000. CrossTalk, The Journal of Defense Software Engineering, marec 1999, str. 12-16.
- CIMdata: Program Review of SAP's mySAP Product Lifecycle Management cPDm Program. Ann Arbor (MI), 2001. 44 str.
- CIMdata: Program Review of SAP’s R/3 PDM. Ann Arbor, Michigan, 1999. 51 str.
- CMstat: New PDM Apps Are More Capable at Managing Complex Data Relationships. B.k., PDM Information Center (PDMIC) [URL: http://www.pdmic.com/articles/artcmsta.html], 2002
- CMstat: Future Bright for Configuration Management Technology. B.k., PDM Information Center (PDMIC) [URL: http://www.pdmic.com/articles/artbfcmt.html], 2002
- Cohen Sholom: A Concept of Operations for Product Lines. CrossTalk, The Journal of Defense Software Engineering, marec 2000, str. 18-20.
- Crnkovic Ivica: Why Do Some Mature Organizations Not Use Mature CM Tools? SCM-9 Symposium, Toulouse, France, 1999 : proceedings. Berlin : Springer, 1999, str. 50-65.
- Duhovnik Jože, Tavčar Jože: Elektronsko poslovanje in tehnični informacijski sistemi. Ljubljana, Univerza v Ljubljani, Fakulteta za strojništvo, 2000. 232 str.
- EDS: Unifying your product lifecycle to accelerate time to market and reduce costs. Electronic Data Systems Corporation [URL: http://www.eds.com/products/plm/teamcenter/pdf/tc_overview.pdf], 2002. 16 str.
- EDS: Accelerating new product innovations through proven engineering collaboration. [URL: http://www.eds.com/products/plm/teamcenter/engineering/pdf/TC_Engineering.pdf], 2002. 12 str.

- Efstathiou Andrew: Justification Tools for Enterprise IT Projects. Boston [URL: http://www.sprint.com/whitepapers/Y_Justification_Tools.pdf], 2002. 14 str.
- Estublier Jacky, Favre Jean-Marie, Morat Philippe: Toward SCM/PDM integration? SCM-8 Symposium, Brussels, Belgium, 1998 : proceedings. Berlin : Springer, 1998, str. 75-93.
- Feiler Peter, Downey Grace: Transaction-Oriented Configuration Management: A Case Study. Pittsburgh, Pennsylvania, Software Engineering Institute, Carnegie Mellon University, 1990. 46 str.
- Feiler Peter H.: Configuration Management Models in Commercial Environments. Pittsburgh, Pennsylvania, Software Engineering Institute, Carnegie Mellon University, 1991. 54 str.
- Frey-Pučko Marjeta: Izboljšave procesnega področja “Upravljanje konfiguracij”. Kranj, interna dokumentacija Iskratel, marec 2002. 25 str.
- Fries Bruce: Successful Implementation of Product Data Management Systems. Premier Design Systems [URL: http://www.pdmic.com/articles/artpdmim.html], 1995.
- Gascoigne Bill : PDM: The Essential Technology for Concurrent Engineering. B.k., PDM Information Center (PDMIC) [URL: http://www.pdmic.com/articles/artetfce.html], 1995.
- Ghezzi Carlo, Jazayeri Mehdi, Mandrioli Dino: Fundamentals of Software Engineering. Englewood Cliffs : Prentice Hall, 1991. 573 str.
- Hewlett-Packard, PDMIC: Understanding Product Data Management. B.k., PDM Information Center (PDMIC) [http://www.pdmic.com/undrstnd.html], 12.6.2002.
- Humphrey Watts S.: Managing the Software Process. Reading, Massachusetts : Addison-Wesley Publishing Company, 1989. 494 str.
- Kelly Marion: Configuration Management. Berkshire: McGraw-Hill Book Company Europe, 1996. 220 str.
- Kežmah Boštjan, Heričko Marjana: Komponentne arhitekture programskih rešitev ERP II na primeru SAP. Zbornik šeste konference OTS, Maribor, 2001, str. 184-191.
- Kruchten Philippe: A Rational Development Process. CrossTalk, The Journal of Defense Software Engineering, julij 1996, str. 11-16.
- Leffingwell Dean, Widrig Don: Managing Software Requirements. B.k. : Addison Wesley, 2000. 491 str.
- Lyon David Douglas: Transparent CM, How To Get There. Pittsfield (MA) : Raven Publishing Company, 2001. 207 str.
- Mäkäräinen Minna: Software change management process in the development of embedded software. Technical Research Centre of Finland, ESPOO, 2000. 185 str.
- Marand: Proposal for a Configuration and Change Management Solution. Ljubljana, 2001. 32 str.
- Marco David: Building and Managing the Meta Data Repository. New York : Wiley, 2000. 392 str.
- Marshall Colin: Change Usage. EDS - predstavitev za Iskratel; interna dokumentacija Iskratel, 2002. 30 str.
- McMahon Paul E.: Distributed Development: Insights, Challenges, and Solutions. CrossTalk, The Journal of Defense Software Engineering, november 2001, str. 4-9.
- MERANT: Assessing ROI for Enterprise Change Management Systems. B.k., QWP01ECM141, 2001. 9 str.
- Milewski Bartosz: Distributed Source Control System. SCM-7 workshop, Boston, MA, 1997 : proceedings. Berlin : Springer, 1997, str. 98-105.

- Neuendorf Steve: Real Process Improvement. CrossTalk, The Journal of Defense Software Engineering, september 1998, str. 20-23.
- Omlie Randall M.: The business case for CM process improvement. Minneapolis, Minnesota, 2000. 33 str.
- Pivka Marjan: Kakovost v programskem inženirstvu. Izola : DESK, 1996. 283 str.
- Probasco Leslee: The Ten Essentials of RUP; The Essence of an Effective Development Process. Rational Software [URL: http://www.rational.com/media/whitepapers/TP177.pdf], Canada 2000. 12 str.
- Product-data management. Department of Trade and Industry [URL: http://www.dti.gov.uk/mbp/bpgt/m9nf75001/m9nf750011.html], 1997.
- Robinson Scott: Understanding BAPI: The most powerful tool in the SAP consultant's kit [URL: http://www.techrepublic.com/printerfriendly.jhtml;jsessionid=1YYI101OM021FTQQACQSFFI?id=r00720021101bal01.htm&rcode=]. CNET Networks, 2002.
- Roubine Olivier: Ogrodje objektnega procesnega modela – Rational Unified Process. Zbornik šeste konference OTS, Maribor, 2001, str. 22-31.
- Shore Barry: Size and Culture as Determinants of IT Strategy in International Supply Chain Management. Zbornik konference 'Challenges of Information Technology Management in the 21st Century', Anchorage, Alaska. Idea Group Publishing, London 2000, str. 148-150.
- Sorensen Reed: CCB - An Acronym for "Chocolate Chip Brownies"?. CrossTalk, The Journal of Defense Software Engineering, marec 1999, str. 3-6.
- Stevens Richard: Requirements through the e-business lifecycle. B.k., [URL: http://www.telelogic.com/webinars/archive.cfm], 2000. 54 str.
- Stevens Richard, Martin James: What is requirements management. B.k. [URL: http://www.telelogic.com/industries/financial/papers/index.cfm], 12.4.2002. 13 str.
- Šmid Marko: Integracija - ključ do uspešnega e-poslovanja. Šesta konferenca OTS, Maribor, 2001, str. 192-198.
- Taborda Louis : Configuration Management Evolution. B.k. [URL: http://www.telelogic.com/webinars/archive.cfm], 2001. 49 str.
- Taramaa Jorma: Practical development of software configuration management for embedded systems. Oulu, Technical Research Centre of Finland (VTT), 1998. 147 str.
- Westfechtel Bernhard, Conradi Reidar: Software Configuration Management and Engineering Data Management): Differences and Similarities. SCM-8 Symposium, Brussels, Belgium, 1998 : proceedings. Berlin : Springer, 1998, str. 95-105.
- White Brian A.: SCM strategies and Rational ClearCase. Upper Saddle River (NJ) : Addison Wesley 2000. 293 str.
- Wiborg Weber Darcy, Vignaud Jean-Louis: A Framework for Managing Component - Based Development. Telelogic white paper, verzija 1.0, Malmö, 2001. 21 str.
- Yphise: Software evaluation report: Application change management for NT and Unix. B.k., 2000. 45 str.

Viri

- Dolenc Janez: Razvojni proces v RD SI2000. Interna dokumentacija Iskratela. 2001. 5 str.
- EDS: Improving automotive supplier profitability by reducing the cost of product development. [URL: http://www.eds.com/products/plm/ind_solutions/pdf/tc_auto_sup_brochure.pdf], 12.4.2002, 8 str.
- Furlan Bojan; Sistem za Obvladovanje Produktnega Razvoja, Analize in Načrtovanja; Opis in navodila za uporabo. Interna dokumentacija Iskratela; ZXX0502AS-ATL-090 . 1999.
- Glossary of Product Data Management Related Terms [URL: http://www.pdmic.com/glossary/index.html]. 2002.
- Gradiva prodajalcev sistemov: predstavitve sistemov, konceptov, procesnih rešitev, tekoča korespondenca preko elektronske pošte. Interna dokumentacija Iskratela. 2001, 2002.
- Gradivo za letno mednarodno konferenco CMII. Minneapolis, Minnesota. 2000.
- IEEE Standards Collection. Software Engineering. 1994 Edition. The Institute of Electrical and Electronics Engineers, 1994.
- Janša Matjaž, Arizanovič Vladimir: Projektno vodenje, planiranje produktov in produktno vodenje v RD SI2000. Interna dokumentacija Iskratela; TPQ021048-TSL-010. 2000. 17 str.
- Metaphase Enterprise 3, ClearCase Integration User's Guide. Structural Dynamics Research Corporation. 2001.
- Novak Roman, Frey-Pučko Marjeta: Analiza razvojnega procesa produktov SI2000. Kranj, interna dokumentacija Iskratel, julij 1999. 55 str.
- Ošabnik Zdene: Analiza migracije funkcionalnosti PDM Sherpa v okolje SAP. Interna dokumentacija Iskratela. 2000. 6 str.
- OVUM Reports. [URL: http://www.ovum.com]. 1998.
- Pahor David, Drobnič Matija: Leksikon računalništva in informatike. Ljubljana : Pasadena, 2002.
- Podlogar Tomislav: Specifikacija tipov zapisov v aplikaciji sopran. Interna dokumentacija Iskratela; ZXX0502LB-ATL. 1997.
- Podlogar Tomislav: SOPRAN ECC - Nadzor sprememb. Interna dokumentacija Iskratela; ZXX0502LB-ATL-060. 1997.
- Prelovšek Uroš: Primerjava ClearCase - Iskratelov SCM. Interna dokumentacija Iskratela. 2000. 17 str.
- Prelovšek Uroš: QM projekt "Upravljanje konfiguracij (analiza podpornih sistemov)". Interna dokumentacija Iskratela. 2000. 52 str.
- Računalniški slovarček: angleško-slovenski, slovensko-angleški. 3. razširjena izdaja, strokovni urednik Matjaž Gams. Ljubljana: Cankarjeva založba, 1993.
- Robnik Ana: Metodologija razvoja programske opreme za telekomunikacijske sisteme. Interna dokumentacija Iskratela; TPQ027241-TSL-030. 2000. 89 str.
- Sheldon Tom: McGraw-Hill Encyclopedia of Networking & Telecommunications. Osborne/McGraw-Hill, Berkeley, CA, 2001. 1447 str.

- Standard SIST ISO 9001:2000 (sl,en). Sistemi vodenja kakovosti - Zahteve. Tretja izdaja. 2000.

Opomba: Dokumenti, ki so last podjetja Iskratel d.o.o., Kranj, oziroma del njegove interne dokumentacije, niso splošno dostopni. Vpogled vanje je moč dobiti z dovoljenjem podjetja.

Priloga A - Odločitveni model

Na naslednji straneh so podani izpisi odločitvenega modela, dobljeni s pomočjo ekspertnega sistema DEX. Za vsakega od obeh najpomembnejših kriterijev, nastopajočih v matriki portfelja, je prikazano:

- drevo kriterijev, pri čemer so podane kratke oznake ob posameznih nastopajočih kriterijih;
- sledi prikaz zalog vrednosti za nastopajoče kriterije;
- prikazane so tudi tabele odločitvenih pravil za sestavljene kriterije.

Drevo kriterijev

Kriterij	Opis
Konkurenčna sposobnost	Ocena konkurenčne sposobnosti sistema
├ **Funkcionalnost**	Funkcionalnosti sistema in njihova kakovost
│ ├ Konfiguracije	Kakovost rešitev na področju upravljanja konfiguracij in sprememb za proizvode
│ ├ **Projekti**	Podpora vodenju in izvajanju projektov
│ │ ├ Nadzor	Podpora načrtovanju in spremljanju poteka projektnih nalog, aktivnosti
│ │ ├ Timsko delo	Podpora timskemu načinu dela
│ │ └ Veliki projekti	Možnost podpore velikih in kompleksnih projektov
│ └ **Proces**	Raven podpore procesu upravljanja kofiguracij in sprememb izdelkov
│ ├ Avtomatizacija	Način in stopnja avtomatizacije izvajanja operacij
│ ├ Vloge	Podpora različnih vlog v procesu
│ └ Modeliranje	Možnosti za modeliranje procesov
├ **Zgradba sistema**	Kako ustrezna je zgradba (arhitektura) sistema
│ ├ Trdoživost	Trdoživost, zanesljivost uporabe sistema, odpornost na napake in okvare
│ ├ Odprtost	Ustrezanje določenim standardom in možnost dodajanja novih modulov
│ └ Varnost podatkov	Zagotavljanje varnosti pri hranjenju in uporabi podatkov
└ **Poslovna uspešnost**	Poslovna uspešnost proizvajalca in samega sistema
├ **Tržna pozicija**	Tržna pozicija proizvajalca in sistema
│ ├ Sistem	Uveljavljenost sistema
│ ├ Proizvajalec	Tržna pozicija in ugled podjetja, ki sistem razvija in vzdržuje
│ └ Perspektiva	Tehnološka in poslovna perspektiva proizvajalca
└ **Podpora**	Raven podpore kupcem oziroma uporabnikom sistema
├ Vpeljava	Pomoč pri vpeljavi sistema (npr. svetovanje, šolanja...)
└ Izboljšave	Stalne izboljšave sistema in tekoče reševanje problemov na sistemu

Zaloge vrednosti

Kriterij	Zaloga vrednosti
Konkurenčna sposobnost	**zelo nizka**; **nizka**; ***visoka***; ***zelo visoka***
Funkcionalnost	**neustrezno**; povprečno; ***dobro***; ***zelo dobro***
Konfiguracije	**slabo**; srednje; ***dobro***; ***zelo dobro***
Projekti	**nesprejemljivo**; ustrezno; ***dobro***; ***zelo dobro***
Nadzor	**slabo**; srednje; ***dobro***
Timsko delo	**neustrezno**; srednje; ***dobro***
Veliki projekti	**neustrezno**; sprejemljivo; ***dobro***
Proces	**nesprejemljivo**; ustrezno; ***dobro***; ***zelo dobro***
Avtomatizacija	**ni ali slabo**; povprečno; ***dobro***
Vloge	**neustrezno**; zadovoljivo; ***dobro***
Modeliranje	**ni ali slabo**; srednje; ***dobro***
Zgradba sistema	**neustrezna**; solidna; ***dobra***; ***napredna***
Trdoživost	**slabo**; srednje; ***dobro***
Odprtost	**slabo**; srednje; ***dobro***
Varnost podatkov	**slabo**; srednje; ***dobro***
Poslovna uspešnost	**slaba**; srednja; ***visoka***; ***zelo visoka***
Tržna pozicija	**slaba**; povprečna; ***dobra***; ***zelo dobra***
Sistem	**neuveljavljen**; uveljavljen; ***zelo uveljavljen***
Proizvajalec	**neuveljavljen**; uveljavljen; ***zelo uveljavljen***
Perspektiva	**ni ali slaba**; srednja; ***dobra***
Podpora	**slaba**; povprečna; ***dobra***; ***zelo dobra***
Vpeljava	**slaba**; povprečna; ***dobra***
Izboljšave	**neustrezno**; povprečno; ***dobro***

Tabele odločitvenih pravil

	Funkcionalnost	Zgradba sistema	Poslovna uspešnost	Konkurenčna sposobnost
1	neustrezno	*	*	zelo nizka
2	*	neustrezna	*	zelo nizka
3	povprečno	solidna	<=srednja	nizka
4	povprečno	>=solidna	slaba	nizka
5	povprečno:dobro	solidna:dobra	slaba	nizka
6	povprečno	solidna:dobra	>=visoka	visoka
7	povprečno	>=solidna	visoka	visoka
8	povprečno:dobro	solidna	>=visoka	visoka
9	povprečno	dobra	>=srednja	visoka
10	povprečno	>=dobra	srednja:visoka	visoka
11	povprečno:dobro	>=dobra	srednja	visoka
12	dobro	solidna	>=srednja	visoka
13	dobro	>=solidna	srednja	visoka
14	>=dobro	solidna	srednja	visoka
15	dobro	napredna	<=srednja	visoka
16	>=dobro	napredna	slaba	visoka
17	zelo dobro	solidna	<=srednja	visoka
18	zelo dobro	>=solidna	slaba	visoka
19	>=povprečno	napredna	zelo visoka	zelo visoka
20	>=dobro	>=dobra	>=visoka	zelo visoka
21	zelo dobro	>=solidna	>=visoka	zelo visoka
22	zelo dobro	>=dobra	>=srednja	zelo visoka

	Konfiguracije	Projekti	Proces	Funkcionalnost
1	slabo	*	*	neustrezno
2	*	nesprejemljivo	*	neustrezno
3	*	*	nesprejemljivo	neustrezno
4	srednje	ustrezno:dobro	>=ustrezno	povprečno
5	srednje	>=ustrezno	ustrezno:dobro	povprečno
6	srednje:dobro	ustrezno	>=ustrezno	povprečno
7	srednje:dobro	>=ustrezno	ustrezno	povprečno
8	>=srednje	ustrezno	ustrezno:dobro	povprečno
9	>=srednje	ustrezno:dobro	ustrezno	povprečno
10	srednje:dobro	zelo dobro	zelo dobro	dobro
11	dobro	>=dobro	>=dobro	dobro
12	>=dobro	dobro	dobro	dobro
13	zelo dobro	ustrezno	zelo dobro	dobro
14	zelo dobro	zelo dobro	ustrezno	dobro
15	zelo dobro	>=dobro	zelo dobro	zelo dobro
16	zelo dobro	zelo dobro	>=dobro	zelo dobro

	Nadzor	Timsko delo	Veliki projekti	Projekti
1	slabo	*	*	nesprejemljivo
2	*	neustrezno	*	nesprejemljivo
3	*	*	neustrezno	nesprejemljivo
4	srednje	>=srednje	>=sprejemljivo	ustrezno
5	>=srednje	srednje	>=sprejemljivo	ustrezno
6	dobro	dobro	sprejemljivo	dobro
7	dobro	dobro	dobro	zelo dobro

	Avtomatizacija	Vloge	Modeliranje	Proces
1	ni ali slabo	*	*	nesprejemljivo
2	*	neustrezno	*	nesprejemljivo
3	*	*	ni ali slabo	nesprejemljivo
4	povprečno	zadovoljivo	>=srednje	ustrezno
5	povprečno	>=zadovoljivo	srednje	ustrezno
6	>=povprečno	zadovoljivo	srednje	ustrezno
7	povprečno	dobro	dobro	dobro
8	dobro	zadovoljivo	dobro	dobro
9	dobro	dobro	srednje	dobro
10	dobro	dobro	dobro	zelo dobro

	Trdoživost	Odprtost	Varnost podatkov	Zgradba sistema
1	slabo	*	*	neustrezna
2	*	slabo	*	neustrezna
3	*	*	slabo	neustrezna
4	srednje	srednje	>=srednje	solidna
5	srednje	>=srednje	srednje	solidna
6	>=srednje	srednje	srednje	solidna
7	srednje	dobro	dobro	dobra
8	dobro	srednje	dobro	dobra
9	dobro	dobro	srednje	dobra
10	dobro	dobro	dobro	napredna

	Tržna pozicija	Podpora	Poslovna uspešnost
1	slaba	*	slaba
2	*	slaba	slaba
3	povprečna	povprečna: dobra	srednja
4	povprečna: dobra	povprečna	srednja
5	povprečna	zelo dobra	visoka
6	dobra	dobra	visoka
7	zelo dobra	povprečna	visoka
8	>=dobra	zelo dobra	zelo visoka
9	zelo dobra	>=dobra	zelo visoka

	Sistem	Proizvajalec	Perspektiva	Tržna pozicija
1	**neuveljavljen**	*	*	**slaba**
2	<=uveljavljen	**neuveljavljen**	**ni ali slaba**	**slaba**
3	uveljavljen	*	srednja	povprečna
4	>=uveljavljen	**neuveljavljen**	>=srednja	povprečna
5	>=uveljavljen	<=uveljavljen	srednja	povprečna
6	uveljavljen	>=uveljavljen	<=srednja	povprečna
7	>=uveljavljen	uveljavljen	<=srednja	povprečna
8	>=uveljavljen	>=uveljavljen	**ni ali slaba**	povprečna
9	***zelo uveljavljen***	**neuveljavljen**	*	povprečna
10	***zelo uveljavljen***	<=uveljavljen	<=srednja	povprečna
11	***zelo uveljavljen***	*	**ni ali slaba**	povprečna
12	uveljavljen	>=uveljavljen	***dobra***	***dobra***
13	***zelo uveljavljen***	***zelo uveljavljen***	srednja	***dobra***
14	***zelo uveljavljen***	>=uveljavljen	***dobra***	***zelo dobra***

	Vpeljava	Izboljšave	Podpora
1	**slaba**	*	**slaba**
2	*	**neustrezno**	**slaba**
3	povprečna	povprečno	povprečna
4	povprečna	***dobro***	***dobra***
5	***dobra***	povprečno	***dobra***
6	***dobra***	***dobro***	***zelo dobra***

Drevo kriterijev

Kriterij	Opis
Zahteve podjetja	Izpolnjevanje zahtev podjetja
Zahteve sistema	Ustrezanje zahtevam Iskratelovega informacijskega sistema
Integracija v sistem	Možnost učinkovite povezave v Iskratelov informacijski sistem
Platforme	Podpora glavnim operacijskim sistemom, ki se uporabljajo pri razvoju izdelkov SI2000
Podporni sistemi	Možnost integracije z drugimi obstoječimi sistemi za podporo poslovanja
Razvojna okolja	Podpora obstoječih razvojnih okolij in aplikacij
Vmesniki	Ustreznost uporabniških vmesnikov
Intuitivnost	Prijaznost, intuitivnost, učinkovitost, uporabniških vmesnikov
Kompletnost	Možnosti raznovrstne uporabe uporabniških vmesnikov
Posebne zahteve	Posebne zahteve procesa in razvojnih metodologij
Modularnost	Podpora delu z moduli in komponentami proizvodov
Zahteve	Možnost učinkovitega upravljanja z zahtevami za proizvode SI2000
Distribuirani razvoj	Podpora razvoja izdelkov na geografsko ločenih lokacijah
Zadovoljstvo	Ocena potencialnega zadovoljstva ožjega in širšega kroga uporabnikov
Ugodnost investicije	Kako ugodna je ta naložba za Iskratel
Višina investicije	Skupen obseg naložbe v podporni sistem
Donosnost	Vrednost kazalnika ROI za obdobje petih let
Ugodnosti	Finančne in druge ugodnosti, ki jih nudi proizvajalec ali dobavitelj sistema
Celovitost rešitve	Celovitost rešitve v smislu izpolnitve namena in dolgoročnosti
Pokritost	Pokritost funkcionalnosti upravljanja konfiguracij in sprememb izdelkov
Uporabnost	Dolgoročnost uporabnosti rešitve (brez večjih posegov)
Partnerstvo	Pričakovana kakovost sodelovanja s proizvajalcem in/ali dobaviteljem
Prehod	Ocena enostavnosti prehoda na novo podporno okolje
Kompleksnost sistema	Zapletenost, težavnost uporabe sistema za različne tipe uporabnikov
Prilagodljivost	Prilagodljivost sistema
Skladnost procesa	Skladnost z zahtevami razvojnega procesa in procesa upravljanja konfiguracij

Zaloge vrednosti

Kriterij	Zaloga vrednosti
Zahteve podjetja	**zelo slabo**; **slabo**; ***dobro***; ***zelo dobro***
Zahteve sistema	**neustrezno**; zadovoljivo; ***dobro***; ***zelo dobro***
Integracija v sistem	**neustrezna**; zadovoljiva; ***dobra***; ***zelo dobra***
Platforme	**HP-UX ali Win**; ***HP-UX in Win***; ***HP-UX, Win, Web***
Podporni sistemi	**slabo**; srednje; ***dobro***
Razvojna okolja	**slabo**; srednje; ***dobro***
Vmesniki	**neustrezni**; povprecni; ***dobri***
Intuitivnost	**neustrezni**; povprečni; ***dobri***
Kompletnost	**enostranski**; večstranski; ***kompletni***
Posebne zahteve	**neustrezno**; zadovoljivo; ***zelo dobro***
Modularnost	**slabo podprto**; povprečno; ***dobro***
Zahteve	**slabo**; srednje; ***dobro***
Distribuirani razvoj	**slabo**; pogojno sprejemljivo; ***dobro***
Zadovoljstvo	**odklonilno**; omahujoče; ***sprejemljivo***; ***zelo visoko***
Ugodnost investicije	**nesprejemljiva**; sprejemljiva; ***zelo ugodna***
Višina investicije	***do 2,1 mio. USD***; 2,1 - 2,8 mio USD; **nad 2,8 mio USD**
Donosnost	**<= 0,8**; **(0,8; 1,0]**; (1,0; 1,2]; ***> 1,2***
Ugodnosti	malo ali nič; ***precej***
Celovitost rešitve	**neustrezna**; zadovoljiva; ***dobra***
Pokritost	**neustrezna**; sprejemljiva; ***dobra***
Uporabnost	**do dveh let**; 2 do 5 let; ***vsaj 5 let***
Partnerstvo	**slabo**; sprejemljivo; ***dobro***
Prehod	**težaven**; srednje; ***enostaven***
Kompleksnost sistema	**kompleksen**; srednje; ***enostaven***
Prilagodljivost	**tog**; delno prilagodljiv; zelo prilagodljiv
Skladnost procesa	**majhna**; delno skladno; ***skladno***

Tabele odločitvenih pravil

	Zahteve sistema	Zadovoljstvo	Prehod	Zahteve podjetja
1	neustrezno	<=*sprejemljivo*	*	zelo slabo
2	neustrezno	*	<=srednje	zelo slabo
3	<=zadovoljivo	odklonilno	težaven	zelo slabo
4	neustrezno	*zelo visoko*	*enostaven*	slabo
5	>=zadovoljivo	odklonilno	>=srednje	slabo
6	zadovoljivo	omahujoče	težaven	slabo
7	>=*dobro*	odklonilno	*	slabo
8	zadovoljivo	>=omahujoče	>=srednje	*dobro*
9	zadovoljivo:*dobro*	>=omahujoče	srednje	*dobro*
10	>=zadovoljivo	omahujoče	>=srednje	*dobro*
11	>=zadovoljivo	omahujoče:*sprejemljivo*	srednje	*dobro*
12	zadovoljivo	>=*sprejemljivo*	*	*dobro*
13	zadovoljivo:*dobro*	>=*sprejemljivo*	<=srednje	*dobro*
14	>=zadovoljivo	*sprejemljivo*	<=srednje	*dobro*
15	>=zadovoljivo	>=*sprejemljivo*	težaven	*dobro*
16	*dobro*	>=omahujoče	<=srednje	*dobro*
17	>=*dobro*	omahujoče	*	*dobro*
18	>=*dobro*	omahujoče:*sprejemljivo*	<=srednje	*dobro*
19	>=*dobro*	>=omahujoče	težaven	*dobro*
20	>=*dobro*	>=*sprejemljivo*	*enostaven*	*zelo dobro*
21	*zelo dobro*	*zelo visoko*	>=srednje	*zelo dobro*

	Integracija v sistem	Vmesniki	Posebne zahteve	Zahteve sistema
1	neustrezna	*	*	neustrezno
2	*	neustrezni	*	neustrezno
3	*	*	neustrezno	neustrezno
4	zadovoljiva	povprecni	>=zadovoljivo	zadovoljivo
5	zadovoljiva	>=povprecni	zadovoljivo	zadovoljivo
6	>=zadovoljiva	povprecni	zadovoljivo	zadovoljivo
7	zadovoljiva:*dobra*	*dobri*	*zelo dobro*	*dobro*
8	*dobra*	>=povprecni	*zelo dobro*	*dobro*
9	>=*dobra*	povprecni	*zelo dobro*	*dobro*
10	*dobra*	*dobri*	>=zadovoljivo	*dobro*
11	>=*dobra*	*dobri*	zadovoljivo	*dobro*
12	*zelo dobra*	*dobri*	*zelo dobro*	*zelo dobro*

	Platforme	Podporni sistemi	Razvojna okolja	Integracija v sistem
1	*	slabo	*	neustrezna
2	*	*	slabo	neustrezna
3	HP-UX ali Win	>=srednje	>=srednje	zadovoljiva
4	<=*HP-UX in Win*	srednje	>=srednje	zadovoljiva
5	<=*HP-UX in Win*	>=srednje	srednje	zadovoljiva
6	*	srednje	srednje	zadovoljiva
7	*HP-UX in Win*	*dobro*	*dobro*	*dobra*
8	*HP-UX, Win, Web*	srednje	*dobro*	*dobra*
9	*HP-UX, Win, Web*	*dobro*	srednje	*dobra*
10	*HP-UX, Win, Web*	*dobro*	*dobro*	*zelo dobra*

	Intuitivnost	Kompletnost	Vmesniki
1	neustrezni	*	neustrezni
2	*	enostranski	neustrezni
3	povprečni	>=večstranski	povprecni
4	>=povprečni	večstranski	povprecni
5	*dobri*	*kompletni*	*dobri*

	Modularnost	Zahteve	Distribuirani razvoj	Posebne zahteve
1	slabo podprto	*	*	neustrezno
2	*	slabo	*	neustrezno
3	*	*	slabo	neustrezno
4	povprečno	>=srednje	>=pogojno sprejemljivo	zadovoljivo
5	>=povprečno	>=srednje	pogojno sprejemljivo	zadovoljivo
6	*dobro*	>=srednje	*dobro*	*zelo dobro*

	Ugodnost investicije	Celovitost rešitve	Partnerstvo	Zadovoljstvo
1	nesprejemljiva	*	*	odklonilno
2	*	neustrezna	*	odklonilno
3	>=sprejemljiva	>=zadovoljiva	slabo	omahujoče
4	sprejemljiva	>=zadovoljiva	>=sprejemljivo	*sprejemljivo*
5	>=sprejemljiva	zadovoljiva	>=sprejemljivo	*sprejemljivo*
6	*zelo ugodna*	*dobra*	>=sprejemljivo	*zelo visoko*

	Višina investicije	Donosnost	Ugodnosti	Ugodnost investicije
1	*	<=(0,8; 1,0]	*	nesprejemljiva
2	nad 2,8 mio USD	*	malo ali nič	nesprejemljiva
3	<=2,1 - 2,8 mio USD	(1,0; 1,2]	*	sprejemljiva
4	*	(1,0; 1,2]	*precej*	sprejemljiva
5	2,1 - 2,8 mio USD	>=(1,0; 1,2]	malo ali nič	sprejemljiva
6	nad 2,8 mio USD	>=(1,0; 1,2]	*precej*	sprejemljiva
7	*do 2,1 mio. USD*	*> 1,2*	*	*zelo ugodna*
8	<=2,1 - 2,8 mio USD	*> 1,2*	*precej*	*zelo ugodna*

	Pokritost	Uporabnost	Celovitost rešitve
1	**neustrezna**	*	**neustrezna**
2	*	**do dveh let**	**neustrezna**
3	>=sprejemljiva	2 do 5 let	zadovoljiva
4	>=sprejemljiva	***vsaj 5 let***	***dobra***

	Kompleksnost sistema	Prilagodljivost	Skladnost procesa	Prehod
1	**kompleksen**	<=delno prilagodljiv	*	**težaven**
2	**kompleksen**	*	<=delno skladno	**težaven**
3	<=srednje	**tog**	*	**težaven**
4	*	**tog**	<=delno skladno	**težaven**
5	*	*	**majhna**	**težaven**
6	<=srednje	zelo prilagodljiv	***skladno***	srednje
7	srednje	>=delno prilagodljiv	>=delno skladno	srednje
8	>=srednje	>=delno prilagodljiv	delno skladno	srednje
9	***enostaven***	**tog**	***skladno***	srednje
10	***enostaven***	>=delno prilagodljiv	***skladno***	***enostaven***

Priloga B - Rezultati vrednotenja

V tej prilogi so podani rezultati vrednotenja, dobljeni z uporabo postavljenega odločitvenega modela. Zatem so za ilustracijo dodane grafične predstavitve vrednotenja nekaterih ključnih kriterijev:

- konkurenčna sposobnost;
- funkcionalnost sistema;
- zahteve podjetja;
- integracija v sistem.

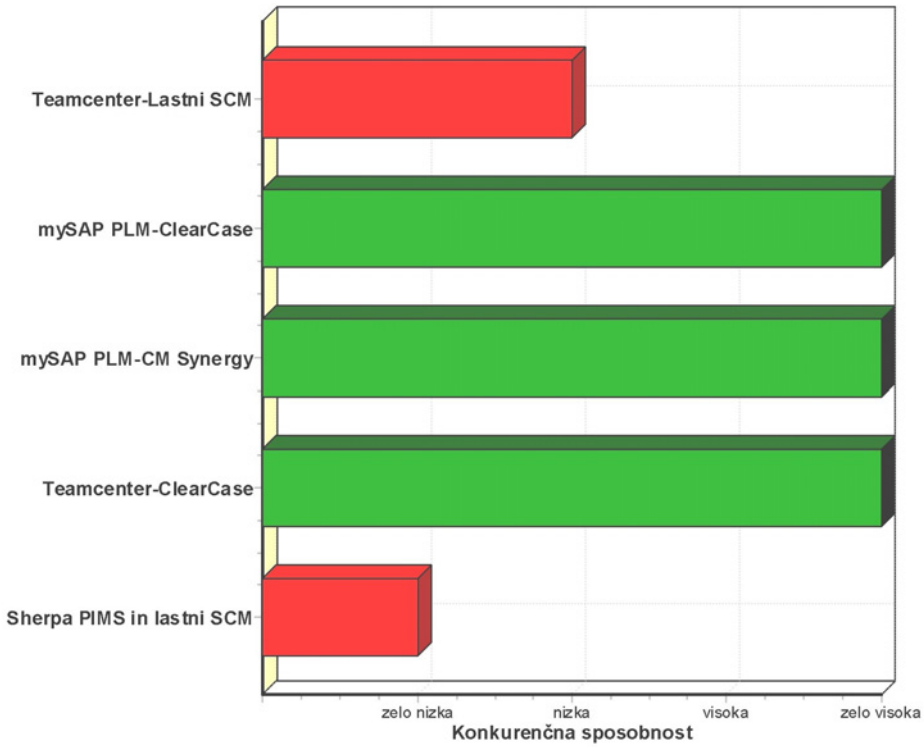
Rezultati vrednotenja

Kriterij	Sherpa PIMS in lastni SCM	Teamcenter-ClearCase	mySAP PLM-CM Synergy	mySAP PLM-ClearCase	Teamcenter-Lastni SCM
Konkurenčna sposobnost	**zelo nizka**	***zelo visoka***	***zelo visoka***	***zelo visoka***	**nizka**
Funkcionalnost	**neustrezno**	***dobro***	***dobro***	***dobro***	povprečno
Konfiguracije	srednje	***dobro***	***dobro***	***dobro***	srednje
Projekti	ustrezno	***zelo dobro***	***zelo dobro***	***zelo dobro***	ustrezno
Nadzor	srednje	***dobro***	***dobro***	***dobro***	***dobro***
Timsko delo	srednje	***dobro***	***dobro***	***dobro***	srednje
Veliki projekti	sprejemljivo	***dobro***	***dobro***	***dobro***	sprejemljivo
Proces	**nesprejemljivo**	***dobro***	***zelo dobro***	***dobro***	***dobro***
Avtomatizacija	povprečno	***dobro***	***dobro***	***dobro***	***dobro***
Vloge	zadovoljivo	***dobro***	***dobro***	***dobro***	***dobro***
Modeliranje	**ni ali slabo**	srednje	***dobro***	srednje	srednje
Zgradba sistema	solidna	***dobra***	***napredna***	***napredna***	solidna
Trdoživost	srednje	***dobro***	***dobro***	***dobro***	srednje
Odprtost	srednje	srednje	***dobro***	***dobro***	srednje
Varnost podatkov	***dobro***	***dobro***	***dobro***	***dobro***	***dobro***
Poslovna uspešnost	**slaba**	***visoka***	***visoka***	***visoka***	srednja
Tržna pozicija	**slaba**	***zelo dobra***	***zelo dobra***	***zelo dobra***	povprečna
Sistem	**neuveljavljen**	***zelo uveljavljen***	***zelo uveljavljen***	***zelo uveljavljen***	uveljavljen
Proizvajalec	**neuveljavljen**	***zelo uveljavljen***	***zelo uveljavljen***	***zelo uveljavljen***	uveljavljen
Perspektiva	**ni ali slaba**	***dobra***	***dobra***	***dobra***	srednja
Podpora	***dobra***	povprečna	povprečna	povprečna	povprečna
Vpeljava	***dobra***	povprečna	povprečna	povprečna	povprečna
Izboljšave	povprečno	povprečno	povprečno	povprečno	povprečno

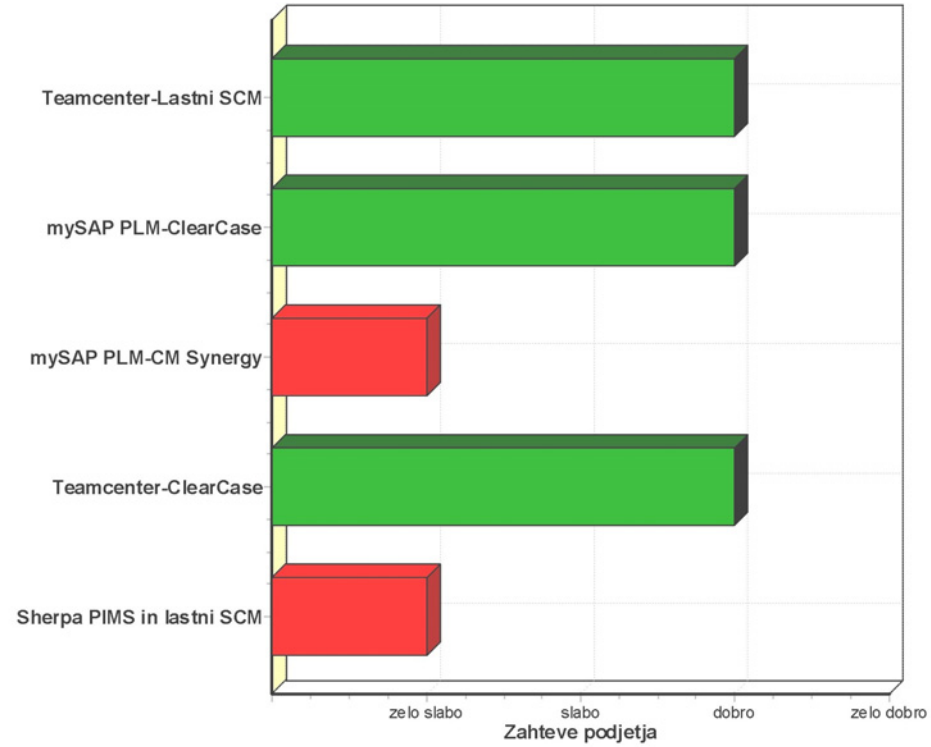
Rezultati vrednotenja

Kriterij	Sherpa PIMS in lastni SCM	Teamcenter-ClearCase	mySAP PLM-CM Synergy	mySAP PLM-ClearCase	Teamcenter-Lastni SCM
Zahteve podjetja	**zelo slabo**	***dobro***	**zelo slabo**	***dobro***	***dobro***
Zahteve sistema	**neustrezno**	zadovoljivo	zadovoljivo	zadovoljivo	zadovoljivo
Integracija v sistem	zadovoljiva	***dobra***	***zelo dobra***	***zelo dobra***	zadovoljiva
Platforme	**HP-UX ali Win**	***HP-UX, Win, Web***	***HP-UX, Win, Web***	***HP-UX, Win, Web***	**HP-UX ali Win**
Podporni sistemi	srednje	srednje	***dobro***	***dobro***	srednje
Razvojna okolja	srednje	***dobro***	***dobro***	***dobro***	srednje
Vmesniki	**neustrezni**	povprecni	povprecni	povprecni	povprecni
Intuitivnost	**neustrezni**	povprečni	povprečni	povprečni	povprečni
Kompletnost	večstranski	večstranski	večstranski	***kompletni***	večstranski
Posebne zahteve	**neustrezno**	zadovoljivo	zadovoljivo	zadovoljivo	zadovoljivo
Modularnost	***dobro***	povprečno	povprečno	povprečno	***dobro***
Zahteve	srednje	***dobro***	***dobro***	***dobro***	***dobro***
Distribuirani razvoj	**slabo**	pogojno sprejemljivo	pogojno sprejemljivo	pogojno sprejemljivo	pogojno sprejemljivo
Zadovoljstvo	**odklonilno**	***sprejemljivo***	**odklonilno**	***sprejemljivo***	***sprejemljivo***
Ugodnost investicije	**nesprejemljiva**	sprejemljiva	**nesprejemljiva**	sprejemljiva	sprejemljiva
Višina investicije	***do 2,1 mio. USD***	2,1 - 2,8 mio USD	**nad 2,8 mio USD**	**nad 2,8 mio USD**	***do 2,1 mio. USD***
Donosnost	**(0,8; 1,0]**	(1,0; 1,2]	***> 1,2***	***> 1,2***	(1,0; 1,2]
Ugodnosti	malo ali nič	***precej***	malo ali nič	***precej***	malo ali nič
Celovitost rešitve	**neustrezna**	***dobra***	***dobra***	***dobra***	zadovoljiva
Pokritost	sprejemljiva	***dobra***	***dobra***	***dobra***	sprejemljiva
Uporabnost	**do dveh let**	***vsaj 5 let***	***vsaj 5 let***	***vsaj 5 let***	2 do 5 let
Partnerstvo	***dobro***	sprejemljivo	sprejemljivo	sprejemljivo	sprejemljivo
Prehod	srednje	**težaven**	**težaven**	**težaven**	srednje
Kompleksnost sistema	srednje	srednje	**kompleksen**	srednje	srednje
Prilagodljivost	zelo prilagodljiv	delno prilagodljiv	delno prilagodljiv	delno prilagodljiv	delno prilagodljiv
Skladnost procesa	***skladno***	**majhna**	**majhna**	**majhna**	delno skladno

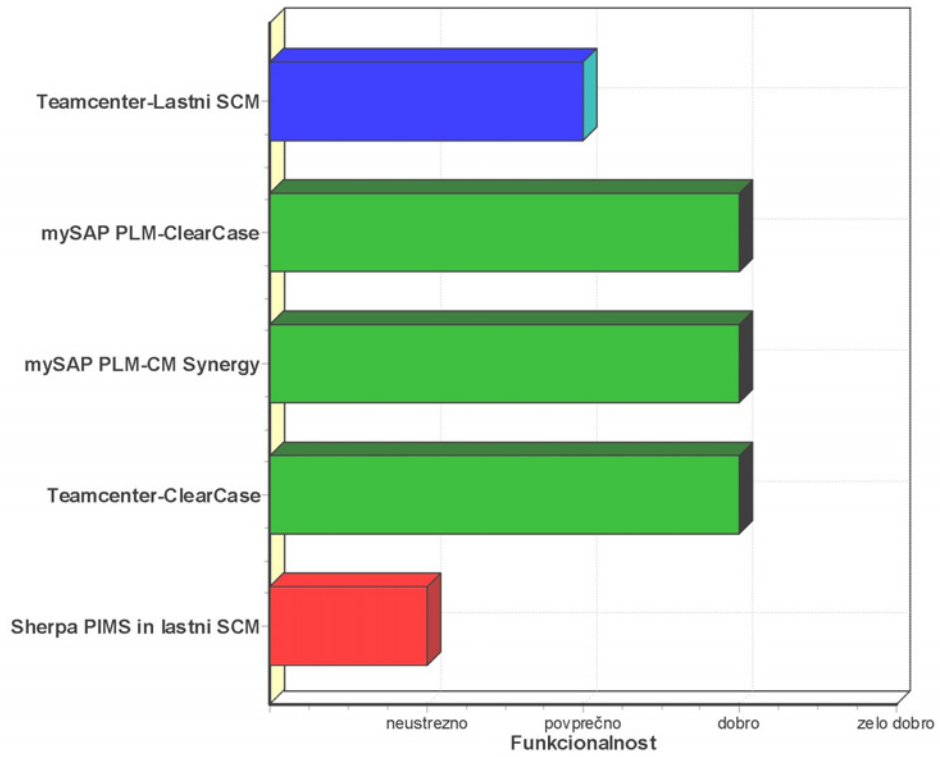
Grafikon



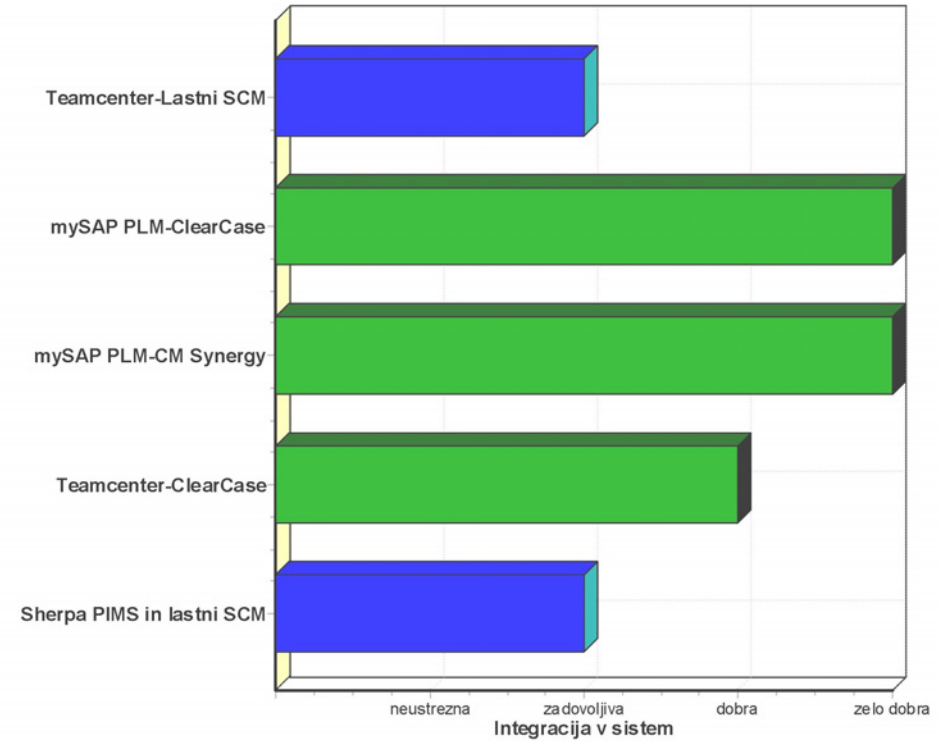
Grafikon



Grafikon



Grafikon



Priloga C - Razlaga pojmov, kratic

ABAP. SAP-ov jezik četrte generacije.

Alert. Tudi: alarm, opozorilnik. V grafičnih uporabniških okoljih opozorilo, ki se, če nastane napaka (npr. pomanjkanje pomnilnika), pokaže v posebnem pogovornem oknu na zaslonu in navadno zahteva od uporabnika odziv oziroma poseg. Pogosto ga spremlja zvočni signal.

ANSI. Kratica za American National Standards Institute - organizacija, ki določa standarde kodiranja in signalizacijske sheme ter predstavlja ZDA v združenju ISO.

Arhitektura. Glej: Zgradba.

ASCII. Kratica za American Standard Code for Information Interchange - ameriški standardni nabor znakov za izmenjavo informacij. Sedembitni nabor znakov, ki je danes v splošni rabi.

BAPI. Kratica za Business Application Programming Interfaces - SAP-ov mehanizem za izmenjavo podatkov z drugimi sistemi za podporo poslovanju.

BOD. Kratica za Bill of Documents - sestavnica, seznam ali kosovnica, ki jo na najnižjem nivoju sestavljajo gradniki dokumentacije.

BOM. Kratica za Bill of Materials. Sestavnica (lahko tudi: seznam uporabljenega gradiva ipd.), določena s seznamom kosovnic, zbirk ali posameznih materialnih delov, ki sestavljajo obravnavani izdelek ali njegovo komponento.

BOR. Kratica za Business Object Repository. Odlagališče poslovnih objektov, jedro sistema SAP R/3. Njegova vsebina določa model poslovanja podjetja.

BP. Kratica za "baza podatkov".

CAD. Kratica za Computer-Aided Design (računalniško podprto načrtovanje). Računalniški sistemi in programje za načrtovanje, oblikovanje in modeliranje izdelkov v elektrotehniki, strojništvu ipd.

CAM. Kratica za Computer-Aided Manufacturing (računalniško podprta proizvodnja). Skupen izraz za uporabo računalnikov pri avtomatizaciji proizvodnje, tako pri izdelavi in sestavljanju izdelkov kot tudi pri nadziranju kakovosti.

CASE. Kratica za Computer Assisted Software Engineering (računalniško podprto inženirstvo programja). Nabor programskih orodij in postopkov za podporo programskemu inženirstvu v vseh fazah razvojnega cikla programske opreme.

CCB. Kratica za Configuration Control Board - odbor za nadzor konfiguracij. Glavni namen teh odborov je pregledati vpliv predlaganih sprememb na zasnovo oziroma načrt proizvoda. Tudi: kratica za Change Control Board - odbor za nadzor sprememb. Njihov namen se v osnovi ujema z namenom odborov za nadzor konfiguracij.

CM. Kratica za Configuration Management. Upravljanje konfiguracij obravnavanih ciljnih objektov - t.s. običajno proizvodi in/ali njihove komponente (npr. programska oprema).

CMII. Model za upravljanje konfiguracij, ki ga je razvil institut ICM. Namen modela je na podlagi najboljših izkušenj zagotoviti tak nivo procesnega področja upravljanja konfiguracij, ki bo poslovnemu sistemu omogočal prehod na drugi nivo modela CMM. Tudi: istoimensko združenje.

CMM. Kratica za Capability Maturity Model. CMM je model, ki ga je razvil inštitut CMU/SEI (t.j. Inštitut za programsko iženirstvo univerze Carnegie Mellon iz Pittsburgha) in je namenjen za ocenjevanje zrelosti procesa izdelave in vzdrževanja programske opreme.

CM sistem. Sinonim za sistem za upravljanje konfiguracij izdelkov ali za upravljanje konfiguracij katere od njihovih komponent, npr. programske opreme. Sinonim tako za SCM kot za PDM sistem oziroma za kakršen koli sistem, katerega ključna funkcija je tudi upravljanje konfiguracij

CPM. Kratica za Collaborative Product Management - upravljanje proizvodov s sodelovanjem. Industrijski koncept podjetja EDS, vgrajen v sistem Teamcenter. Njegov namen je v združevanju geografsko razpršenih "otokov znanja o proizvodu" v enoten vir veljavnih informacij o proizvodu skozi ves njegov življenjski cikel.

CRM. Kratica za Customer Relationship Management - upravljanje odnosov s kupci. Industrijska disciplina, ki obravnava odnose z naročniki, kupci. Ta proces je lahko avtomatiziran s t.im. CRM sistemi.

Delovni proizvod. Delovni proizvod je dokument, datoteka, podatki – skratka vsak vmesni produkt, ki nastane med izvajanjem določenega procesa.

DMS. Sherpin jezik, podoben SQL (Structured Query Language), z določenimi razširitvami, ki omogočajo delo z zapisi Sherpine baze podatkov.

Dokumentacija. Zbirka dokumentov o določeni zadevi oziroma vsaka napisana ali slikovna informacija, ki opisuje, določa, poroča ali potrjuje aktivnosti, zahteve, postopke ali rezultate.

ECN. Kratica za Engineering Change Notice - ocena inženirske spremembe. Z ECN-om uporabnik določi spremembo na produktu in aktivnosti, potrebne za izvedbo spremembe.

ECO. Kratica za Engineering Change Order - nalog za izvedbo inženirske spremembe. Ustvari se na podlagi podanega ECR-a. Običajno predstavlja delovni nalog za izvedbo spremembe.

ECR. Kratica za Engineering Change Request - zahteva za inženirsko spremembo. Z ECR-om prijavitelj poda zahtevo za spremembo proizvoda ali njegove komponente.

EIA. Kratica za Electronic Industries Association (industrijsko združenje).

Gradnik konfiguracije. Skupek materialne ali programske opreme (ali obeh skupaj), ki se obravnava kot enotna entiteta.

HP-UX. Oznaka za različico operacijskega sistema Unix, ki jo je razvilo podjetje Hewlett-Packard.

HTML. Kratica za Hypertext Markup Language. Jezik za izdelavo in oblikovanje spletnih strani v Web okolju.

ICM. Kratica za Institute of Configuration Management - tu je bil razvit model CMII.

Idocs. SAP-ov mehanizem za prenos in izmenjavo podatkov med različnimi moduli tega sistema.

IEEE. Kratica za Institute of Electrical and Electronic Engineers - neprofitno združenje s sedežom v Združenih državah Amerike, ki med drugim razvija standarde s področij podatkovnih komunikacij in programskega inženirstva.

Inspekcija. Statična analitična tehnika, ki temelji na vizualnem pregledovanju razvojnih produktov z namenom odkrivanja napak, kršenja razvojnih standardov in ostalih problemov.

Instanca. Konkreten primerek, 'pojavek' nekega razreda, tipa...

ISO. Kratica za International Organization for Standardization - svetovna zveza teles za izdelavo nacionalnih standardov. V njej so predstavniki iz več kot 100 držav.

IT. Kratica za "informacijska tehnologija" (angl. Information Technology)

ITWE. Naziv ene od ključnih poslovnih enot v podjetju Iskratel, d.o.o., Kranj.

Izhodiščna konfiguracija. Pojem se lahko nanaša tako na celoten industrijski proizvod kot na njegov sestavni del, komponento. Gre za poseben primer osnovnice. Ko se dana konfiguracija oziroma kosovnica proizvoda (ali komponente) določi (postavi, sestavi, 'zamrzne', fiksira), formalno pregleda in potrdi, služi kot izhodišče, osnova za nadaljnji razvoj tega proizvoda (ali komponente). Spreminja se lahko le skozi formalne postopke, predvidene v procesu sprememb.

Izpeljani objekt; tudi: derivirani objekt. Datoteka (vključno z njenimi metapodatki, kot jih vodi operacijski ali podporni sistem), ki nastane kot vmesni ali končni produkt pri gradnji programske opreme. Običajno jih sistem hrani z namenom pouporabe in prihranitve časa pri ponovni izgradnji. Za razliko od (navadnih) gradnikov konfiguracije se izpeljani objekti običajno ne hranijo v knjižnicah, temveč v delovnih okoljih in so začasne narave. Ko niso več aktualni, jih uporabnik ali sam sistem pobriše iz datotečne mape.

Izvorna koda. Računalniške instrukcije in definicije podatkov, izražene v obliki, primerni kot vhod za zbirnik, prevajalnik ipd. Primer: Izvorni program je narejen iz izvorne kode.

Kompatibilnost. Zmožnost, da dva ali več sistemov izvajata zahtevane funkcionalnosti, medtem ko si delita isto strojno ali programsko okolje. Lahko tudi: zmožnost dveh ali večih sistemov ali komponent za izmenjavo informacij.

Kompleksnost. Stopnja zapletenosti, težavnosti razumevanja in verifikacije načrta oziroma implementacije sistema ali njegove komponente.

Komponenta. En od delov, ki sestavljajo sistem. Komponenta je lahko ali strojna/materialna ali programska in se lahko deli naprej v druge komponente. Opomba: Pojmi, kot so "modul", "komponenta" ali "enota" se običajno uporabljajo kot sinonimi, ali so eni določeni kot podelementi drugih, na različne načine, odvisno od konteksta. Povezave med temi pojmi še niso dokončno standardizirane. V Feilerjevi teoriji konceptualnih modelov upravljanja konfiguracij je komponenta sinonim za gradnik konfiguracije.

Konsistentnost. Stopnja enotnosti, standardizacije in neodvisnosti z ozirom na možnost pojavljanja protislovij med dokumenti, kosovnicami ali gradniki konfiguracije.

Kosovnica; zbirka. Kosovnica je lahko določena z gradniki ali drugimi kosovnicami in je lahko sestavni del konfiguracije celotnega proizvoda ali njegove komponente. V podobne namene se uporabljajo zbirke, sestavnice ipd.

Mejnik (angl. milestone). Načrtovana točka, za katero je odgovoren nek posameznik in se uporablja za merjenje napredovanja.

Metapodatki. To so vsi fizični podatki (vsebovani v programski opremi in drugih medijih) in znanje, ki ga premorejo zaposleni ali je shranjeno na različnih medijih, znotraj ali zunaj organizacije, vključno z informacijami o fizičnih podatkih, tehničnih in poslovnih procesih, pravilih in omejitvah podatkov ter podatkovnih strukturah, ki jih uporablja organizacija. Tudi: informacije o gradnikih in podatkih, ki so pod kontrolo sistema za upravljanje podatkov o izdelkih ali sistema za upravljanje konfiguracij izdelkov.

MRP, MRP-II, ERP, EDM. MRP (angl. Materials Requirements Planning) je najprej označeval t.im. področje planiranja zahtev za za materialno opremo, pri čemer je bil poudarek na sestavnicah, vključenih v proizvod, dostavljen naročniku. Pri tem je bilo razmeroma malo poudarka na procesu pretvorbe 'surovin' v končne izdelke oziroma storitve. Zato je bil koncept nadgrajen z MRP-II, t.j. planiranjem virov za proizvodnjo (angl. Manufacturing Resource Planning). V podporo temu so nastali veliki podporni sistemi, katerih funkcije so vključevale načrtovanje in spremljanje omejenih virov, načrtovanje in spremljanje proizvodnje, podporo pri finančnem poslovanju, ne nazadnje pa tudi nadzor sprememb nad konfiguracijo proizvodov. Bolj ko ta podpora sega na področja izven proizvodnje in bolj ko pokriva področje poslovanja s proizvodi na nivoju celotnega podjetja, bližje smo t.im. ERP (kratica za Enterprise Resource Planning, planiranje virov podjetja) sistemom in EDM (kratica za Enterprise Data Management, upravljanje s podatki podjetja) sistemom.

MVFS. Kratica za MultiVersioned File System. Transparentni datotečni sistem SCM orodja Rational ClearCase. Namenjen je celovitemu nadzoru pri delu z verzijami gradnikov konfiguracije (t.im. elementov).

Nadzor konfiguracije. Poddisciplina upravljanja konfiguracij, ki sestoji iz ovrednotenja, usklajevanja, potrjevanja/zavračanja in izvedbe sprememb konfiguracijskih gradnikov, po formalno opravljenem poistovetenju konfiguracije. Sinonim za nadzor sprememb.

Nadzorni odbor. Skupina ljudi, odgovornih za ovrednotenje in potrjevanje/zavračanje predlaganih sprememb konfiguracijskih gradnikov. Poleg tega je njihova naloga tudi v tem, da zagotovijo izvedbo potrjenih sprememb.

Nadzor verzij, kontrola verzij. Vzpostavitev številčnega stanja za vsako datoteko in gradnik konfiguracije znotraj nadzorovanega območja ter povečanje tega stanja, kadarkoli se datoteka oziroma konfiguracijski gradnik spremeni. Nadzor verzij vsebuje tudi možnost sledenja razlik med verzijami in je tesno povezan z nadzorom doseganja gradnikov.

Naročnik. Naročnik je posameznik ali neka druga entiteta, za katero se razvija proizvod, oziroma bodoči uporabnik proizvoda. Naročnik je lahko vnaprej znan, v tem primeru se je z njim možno pogajati.

NASA. Kratica za National Aeronautics and Space Administration - ameriška vesoljska agencija.

NFS. Kratica za Network File System - porazdeljena datotečna storitev v okoljih tipa strežnik/odjemalec, ki zagotavlja transparentno doseganje datotek. Razvit v osemdesetih letih prejšnjega stoletja pri Sun Mycrosistems. Danes predstavlja odprt internetni protokol.

Objektna koda. Računalniški podatki in definicije podatkov v obliki, ustrezajoči rezultatu obdelave z zbirnikom ali prevajalnikom.

OMG. Kratica za Object Management Group - organizacija, ustanovljena 1989, katere namen je promocija objektnih tehnologij na porazdeljenih računalniških sistemih. Združuje preko 700 končnih uporabnikov in podjetij, ki razvijajo ali prodajajo programsko opremo.

Osnovnica (angl. baseline). Sinonim za formalno pregledano in potrjeno specifikacijo ali sam proizvod. Osnovnica služi kot izhodišče oziroma temelj za nadaljnji razvoj. Spreminja se lahko le skozi formalne postopke, predvidene v procesu sprememb. Lahko tudi: dokument ali množica dokumentov, formalno razvit in fiksiran v določeni časovni točki v svojem življenjskem ciklu. Opomba: podana osnovnica proizvoda in potrjene spremembe, izhajajoče iz te osnovnice, določajo (identificirajo) trenutno konfiguracijo proizvoda.

Patch. Sprememba, izvedena neposredno na objektnem programu, brez ponovnega prevajanja izvornega programa. Lahko tudi: Sprememba, narejena na izvornem programu, kot popravek napake v zadnjem trenutku ali kot posledica dodatnega domisleka.

PDM (Product Data Management) in novejši, iz PDM in PIM izpeljani koncepti: CPC, PDC, CPM, cPDm, PLM, ePLM ipd. Upravljanje s podatki o proizvodih. Najvišja stopnja avtomatizacije proizvoda in procesa, ki jo lahko dosežemo z ozirom na finančna sredstva, tehnologijo, kulturno okolje in organizacijsko zavezanost podjetja. V zadnjih letih so se pojavili novi akronimi, da bi ponazorili večji obseg storitev, ki pomagajo celovitemu upravljanju življenskega cikla proizvoda v virtualnem podjetju. To so npr. CPC (Collaborative Product Commerce), PDC (Product Definition and Commercialization), CPM (Collaborative Product Management), cPDm (collaborative Product Definition management), PLM (Product Lifecycle Management) in ePLM (electronic Product Lifecycle Management).

PIM. Kratica za Product Information Management. Upravljanje z informacijami o proizvodih. Običajno se uporablja kot sinonim za PDM in opisuje polno integracijo vseh informacij o proizvodih skozi njihov življenjski cikel.

Platforma. Operacijski sistem na določenem računalniku ali v določeni računalniški mreži.

PO. Kratica za "programska oprema".

Podpora gradnji programske opreme, nadzor nad gradnjo. Podporne funkcije, ki omogočajo karseda učinkovito gradnjo programske opreme iz osnovnih, izvornih gradnikov konfiguracije. Primeri: funkcije orodja make, vodenje metapodatkov za gradnike in izpeljane gradnike, pouporaba izpeljanih gradnikov ipd.

Pogled (angl. view). Uporabnikovo delovno okolje pri delu s sistemom ClearCase.

Poistovetenje (identificiranje) konfiguracije. Element upravljanja s konfiguracijo. Sestavljen je iz izbire konfiguracijskih gradnikov za sistem ter zabeleženjem njihovih funkcionalnih in fizičnih lastnosti v tehnični dokumentaciji.

Ponovljivost. Zmožnost reproduciranja gradnika konfiguracije, izhodiščne konfiguracije ali kakršne koli druge entitete ali aktivnosti, ki nastopa v procesu, natančno v takšni obliki, kot je bila v določeni časovni točki ali v določeni izdaji. Pod ponovljivostjo se razume, da zagotavlja tudi možnost preverjanja, da je bila reprodukcija ustrezno izvedena.

PRB. Kratica za Problem Resolution Board - odbor za razrešitev problemov. Namen PRB je identificirati in razrešiti probleme, ki se pojavijo v fazah proizvodnje in vzdrževanja izdelkov.

Pregled kode (angl. code review). Sestanek, na katerem se programska koda predstavi projektnemu vodstvu, produktnim vodjem, uporabnikom, kupcem ali ostalim zainteresiranim stranem, z namenom, da jo ti komentirajo ali potrdijo.

Prilagodljivost za večje projekte/produkte (angl. scaleability). Sposobnost sistema, da funkcionira tudi tedaj, ko obseg virov, produktov in aktivnosti v projektu naraste.

Proces. Proces je nabor aktivnosti, ki se izvajajo za dosego nekega določenega cilja. Aktivnosti znotraj procesa se lahko izvajajo zaporedno, rekurzivno ali vzporedno. Dobro določen proces vključuje aktivnosti, za katere so vstopni in izstopni delovni proizvodi natančno definirani, in za katere je vzpostavljen mehanizem nadzora učinkovitosti.

Proces razvoja proizvoda (ali njegove komponente, npr. programske opreme). Proces, pri katerem so potrebe uporabnikov prevedene v (programski) proizvod. Proces vključuje: prevedbo uporabnikovih potreb v zahteve za (programski) izdelek, prevedbo teh zahtev v načrt (angl. design), izvedbo načrta, testiranje izdelka, včasih pa tudi namestitev in testiranje za operativno delovanje na terenu. Opomba: te aktivnosti se lahko prekrivajo ali pa se izvajajo zaporedno.

Programska oprema (angl. software). Računalniški programi, postopki in pripadajoča dokumentacija, nanašajoči se na delovanje računalniškega sistema.

Proizvod; izdelek; produkt. Izdelek podjetja v svoji obliki in sestavi skozi ves svoj življenjski cikel, od zamisli do umika s tržišča. Navedeni pojmi se uporabljajo kot sinonimi.

Projekt. Projekt je skupek aktivnosti in virov, katerih cilj je razvoj, proizvodnja ali vzdrževanje določenega produkta. Projekt običajno razpolaga s svojimi sredstvi, ima ločeno evidenco stroškov in lasten časovni načrt.

Projektni program. Množica projektov, povezana s skupnim ciljem ali s skupnim izdelkom oziroma družino podobnih izdelkov. Primeri: vesoljski program Apollo, projektni program v5 SI2000 itd.

RCS. Kratica za Revision Control System. Enostaven sistem za nadzor verzij programskih gradnikov. Po funkcijah soroden orodju SCCS.

RDBMS. Kratica za Relational Database Management System - sistem za upravljanje relacijskih zbirk podatkov.

RFC. Kratica za Remote Function Call - funkcijska koda, s katero so realizirani poslovni objekti, predstavljeni z BAPI-ji, v sistemu mySAP.com.

ROI. Kratica za Return on Investment - finančni kazalnik za ocenjevanje donosnosti naložbe (investicije). Količnik, ki pove, koliko denarnih enot dobimo v zameno za eno denarno enoto, ki smo jo vložili v nov sistem oziroma v celoten projekt vpeljave sistema. Vrednosti pri izračunu se običajno prevedejo na sedanjo vrednost denarja.
RUP. Kratica za Rational Unified Process.
SAE. Kratica za združenje Society of Automotive Engineers.
Samba. Zbirka programov, ki zagotavljajo storitve v zvezi z delitvijo in uporabo računalniških virov.
SAP. Kratica za Systems, Applications and Products in Data Processing - podjetje s sedežem v Walldorfu v Nemčiji. Dobavitelj rešitev s področja elektronskega poslovanja.
SCCS. Kratica za Source Code Control System - orodje, namenjeno nadzoru verzij programskih gradnikov.
SCM. Kratica za Software Configuration Management. Upravljanje s konfiguracijo programske opreme. Običajno se navezuje na naslednja funkcionalna področja: nadzor verzij, podpora delovnih okolij, možnost vzporednega (paralelnega) razvoja, podpora razvoja programske opreme na geografsko porazdeljenih lokacijah, nadzor konfiguracije, nadzor sprememb, podpora gradnji in izdaju PO, upravljanje procesa. Tudi: kratica za Supply Change Management, upravljanje z oskrbovalno verigo.
Sestavnica. Glej: BOM.
SI 2000. Družina telekomunikacijskih proizvodov podjetja Iskratel, d.o.o., Kranj.
SID. Kratica za "SCCS identifikator". Identifikator verzije v SCCS datoteki.
Sistem. Zbirka komponent, organiziranih za izpolnitev določenih funkcij.
Sledljivost (angl. traceability). Stopnja, do katere je možno ugotoviti povezave med dvema ali več proizvodi razvojnega procesa. Tu gre posebej za proizvode, ki imajo povezave tipa predhodnik-naslednik ter glavni-podrejeni (angl. master-subordinate). Primer: stopnja ustrezanja zahtevam in zasnovi danega programskega proizvoda. Lahko tudi: stopnja, do katere je ugotovljen razlog za obstoj katerega koli elementa v programskem razvojnem proizvodu.
SMB. Kratica za Server Message Blocks. Visokonivojski protokol za uporabo skupnih datotek v operacijskih sistemih Windows in OS/2.
SO. Kratica za "strojna oprema". Kot sinonim za *strojna oprema* se uporablja tudi izraz *materialna oprema*.
SOPRAN, Sopran. Kratica za "Sistem za Obvladovanje Produktnega Razvoja, Analize in Načrtovanja". Iskratelov PDM sistem, s funkcijami za obvladovanje dokumentacije, problemov, kosovnic proizvodov in njihovih komponent itd.
Strojna oprema, materialna oprema (angl. hardware). Elektronski deli in ohišja izdelkov. Fizična oprema, namenjena obdelovanju, shranjevanju ali prenašanju računalniških programov in podatkov.
Struktura. Definicija hierarhičnih odvisnosti in medsebojnih povezav med gradniki v konfiguraciji proizvoda oziroma v samem proizvodu. Primer: množica datotečnih map in podmap.
Transparentni način dela z gradniki konfiguracije. Način, ki ga podpirajo nekateri sistemi za upravljanje konfiguracij izdelkov oziroma programske opreme (npr. Rational ClearCase s pomočjo sistema MVFS), ki nadzirajo tudi uporabnikovo delovno okolje. Uporabniku je omogočeno neposredno delo na strukturi izdelka oziroma na ustreznih gradnikih konfiguracije izdelka.
UML. Krat. za Unified Modeling Language - jezik za modeliranje informacijskih sistemov in programskih entitet. Na abstraktni ravni se uporablja za vizualiziranje, specificiranje, konstruiranje in dokumentiranje znanja o zapletenih programskih sistemih in njihovem namenu.

Upravljanje konfiguracije (angl. Configuration Management). Disciplina, ki uporablja tehnične in administrativne smernice v zvezi s: poistovetenjem in dokumentiranjem funkcionalnih in fizičnih lastnosti gradnikov konfiguracije; nadzorom sprememb teh lastnosti; beleženjem in poročanjem o obdelavi sprememb in tekočem stanju implementacije; presojanjem ujemanja s specificiranimi zahtevami.

Vejitev (angl. branch). Vejitev gradnika konfiguracije je ena od možnosti, s katero sistem za nadzor verzij podpira paralelni razvoj. Vsaka vejitev ima svojo oznako, sestavljena pa je iz različnih verzij gradnika. Poznamo glavno vejo razvoja gradnika ter stranske veje. Nekateri sistemi imajo tudi možnost združevanja (angl. merge) dveh različnih vejitev.

Verzija (angl. version), revizija (angl. revision). Začetna ali ponovna izdaja gradnika konfiguracije. Začetna ali ponovna izdaja dokumenta, v nasprotju z revizijo, ki nastane z izdajanjem spremenjenih strani glede na prejšnjo revizijo.

View. Glej: Pogled.

Vloženi računalniški sistemi (angl. embedded computer systems). Vloženi računalniški sistem je sistem v okviru večjega sistema. Lahko je izdelan na enem samem integriranem vezju. Primeri vloženih sistemov so čipi, ki nadzirajo funkcije avtomobilov, telekomunikacijskih izdelkov ipd. Pri tem zunanji operacijski sistem ni potreben.

VOB. Kratica za Versioned Object Base. ClearCase-ovo skladišče podatkov, nastajajočih v danem projektu izdelave programske opreme. Omogoča nadzor verzij tako za datoteke kot za datotečne mape.

XML. Kratica za Extended Markup Language. Novejši jezik, ki omogoča ločen nadzor nad vsebino in obliko besedila v Web okolju.

Zgradba; tudi: arhitektura (angl. architecture). Organizacijska struktura proizvoda/sistema ali njegove komponente.

Zgrajen objekt. Delujoča verzija sistema ali komponente, ki vsebuje določeno podmnožico lastnosti oziroma funkcionalnosti, kot jih predvideva končni produkt.

Življenjski cikel izdelka (ali programske opreme). Obdobje, ki se začne z zamislijo proizvoda (lahko programskega) in se konča s trenutkom, ko le ta ni več na razpolago za uporabo. Življenjski cikel proizvoda ali programske opreme navadno vsebuje naslednje faze: izdelava koncepta; postavljanje zahtev; načrtovanje; implementacija; testiranje; instalacija in testiranje na terenu; faza delovanja in vzdrževanja; faza umikanja iz obratovanja (v tej fazi se ukinja podpora proizvodu). Te faze se lahko prekrivajo ali pa se izvajajo zaporedno.

Priloga D - Kratka predstavitev razvoja in uporabe upravljanja konfiguracij izdelkov

i. Disciplina

Elementi nadzora procesa in rezultatov načrtovanja so se od pradavnine uporabljali s strani najbolj znanih svetovnih izumiteljev, kot so bili npr. graditelji egipčanskih piramid, Leonardo da Vinci, Henry Ford, Thomas Alva Edison in mnogi drugi (Berlack, 1992, str. 1). V preteklosti je bil poudarek na ročnem, neavtomatiziranem izvajanju aktivnosti (Lyon, 2001, str. 16). Rezultati teh aktivnosti so mnogokrat predstavljali nepotreben balast. Da bi rešili ta problem, se je sčasoma dajal vse večji poudarek razvoju in razumevanju procesa obvladovanja izdelkov skozi njihov življenjski cikel, po drugi strani pa razvoju in uporabi sistemov orodij za vodenje podatkov o proizvodih ali sistemov za upravljanje konfiguracij proizvodov. Ti so na avtomatiziran način podpirali postopke v procesu obvladovanja produktov skozi njihov celotni življenjski cikel. Vse bolj zapletena tehnologija in vse večja zapletenost izdelkov, še posebej na področjih letalske, elektronske in vojaške industrije je pripeljala do potrebe po formalni disciplini, ki bo služila obvladovanju, upravljanju strukture in konfiguracij teh izdelkov.

V letu 1962 je Air France odgovorila na kritične probleme pri nadzoru in koordinaciji načrtovanja njenih letal z izdelavo in objavo standarda za upravljanje konfiguracij, AFSCM 375-1 (Berlack, 1992, str. 11). Ta standard je prekinjal dolgoletno prakso na področju vodenja projektov. Z izboljšavo komuniciranja v projektu, načrtovanje izdelkov kar naenkrat ni bilo več le stvar posameznih inženirjev, temveč je postalo en od sestavnih elementov upravljanja izdelkov. Na naslovni strani standarda je bila objavljena shema, ki je ponazarjala upravljanje konfiguracij kot vezni člen, ki povezuje različne faze življenjskega cikla izdelka v povezan, delujoč sistem.

V naslednjih letih sta se po tem zgledovala predvsem NASA in ameriška vojska, ki sta od tedaj razvili obsežno množico standardov za upravljanje konfiguracij izdelkov za svoje vesoljske in oborožitvene programe. V tem času so imeli omenjeni standardi zelo pomembno vlogo pri razvoju izdelkov. Niso pa še vključevali kakšnih posebnih zahtev za računalniško podporo tovrstnega upravljanja. Vzrok je bil predvsem v tem, da so tedanji izdelki vsebovali razmeroma majhen delež programske opreme - največji poudarek je bil na strojni oziroma materialni opremi.

Pojava, ki sta v zelo veliki meri zaznamovala področje upravljanje konfiguracij v sedemdesetih in osemdesetih letih, sta pomik zanimanja za boljše upravljanje z izdelki (predvsem z vidika upravljanja življenjskega cikla in kakovosti izdelkov) iz vojaške v splošno poslovno sfero in povečanje deleža in pomena programske opreme v različnih izdelkih. Kot posledica obeh pojavov so se v pisanje standardov za področje upravljanja konfiguracij vse bolj vključevale organizacije, ki so znane po standardih z različnih področij: EIA (Electronic Industries Association), IEEE, SAE (Society of Automotive Engineers), ANSI, ISO, itd. Za podjetja s področja telekomunikacij, kamor spada tudi Iskratel, imajo še posebno veliko vlogo standardi, ki jih je izdalo združenje IEEE.

V devetdesetih letih prejšnjega stoletja se je vse bolj uveljavil internet, ki je tudi zelo vplival na področje upravljanja konfiguracij. Za disciplino upravljanja konfiguracij so se s tem pojavili novi izzivi, npr. upravljanje z vsebino spletnih strani.

V najnovejši verziji standarda ISO 9000 iz leta 2000 obstaja kar nekaj elementov kakovosti, ki so tesno povezani z upravljanjem konfiguracij tako končnih kot delovnih proizvodov, npr. v točkah: 4.2.3 Obvladovanje dokumentov; 4.2.4 Obvladovanje zapisov; 7.3.4 Pregled načrtovanja in razvoja; 7.3.7 Obvladovanje sprememb načrtovanja in razvoja; 7.5.3 Identifikacija in sledljivost; 8.2.4 Nadzorovanje in merjenje proizvodov itd.
Upravljanje konfiguracij na področju programske opreme ima svoje mesto tudi v modelu CMM, ki v sodobnih podjetjih za proizvodnjo PO zavzema vse večji pomen. Humphrey je 1990 razvil tezo o zorenju procesa izdelave PO, ki je danes v svetu splošno priznana. Zorenje procesa izdelave PO deli na pet nivojev (Pivka, 1996, str. 18): začetni nivo; ponovljivi nivo; definirani nivo; upravljani nivo; nivo stalnih izboljšav. CMM model temelji na Demmingovi tezi, da le kontroliran proces zagotavlja pričakovane rezultate s sprejemljivimi stroški, roki in kakovostjo (Pivka, 1996, str. 26). Ustrezen nivo upravljanja konfiguracij PO je pogoj za doseganje drugega nivoja (t.im. ponovljivi nivo) v modelu CMM. Kot pogoj za doseganje drugega nivoja se pojavljajo tudi nekatera druga procesna področja, kot npr. upravljanje zahtev; upravljanje projektov za PO; zagotavljanje kakovosti PO, itd. Model CMM ni mednarodni ali nacionalni standard, vendar ga po svetu vse bolj priznavajo.
Z raziskovanjem področja upravljanja konfiguracij se danes ukvarjajo mnogi znani znanstveni in industrijski inštituti, specializirani za področje programskega inženirstva, po vsem svetu. Nekateri od najbolj znanih od njih so CMU SEI (znan tudi po modelu CMM), Institute of Configuration Management (ICM) na državni univerzi v Arizoni, IEEE itd. Nekatere od univerz (npr. Državna univerza v Arizoni) imajo namen v kratkem uvesti študij discipline upravljanja konfiguracij kot poseben predmet v svojem univerzitetnem izobraževalnem programu.
Upravljanje globalnih podjetij postaja vse težje. Podjetja morajo biti sposobna razumeti, predvideti in zadovoljiti želje kupcev bolje, kot to lahko stori konkurenca. Stara filozofija "če bomo razvili tak in tak izdelek, ga bodo prav gotovo kupovali", ni več učinkovita in podjetjem, ki razvijajo izdelke visoke tehnologije, ne zagotavlja več niti preživetja. Kot kažejo izkušnje s trgov visokih informacijskih tehnologij, bo v prihodnje potrebno dati upravljanju zahtev in želja naročnikov še precej večji poudarek (Marco, 2000, str. 18). S tem namenom se je v zadnjih letih močno razvila industrija CRM sistemov (krat. za Customer Relationship Management) za podporo upravljanja odnosov s kupci, podjetja po vsem svetu pa intenzivno uvajajo tovrstne podporne sisteme. V teh in drugih aktualnih gibanjih, tako na poslovnem kot tudi na drugih področjih, lahko najde svojo vlogo in smisel tudi področje upravljanja konfiguracij.

ii. Podporni sistemi

V nadaljevanju je na kratko orisan razvoj najznačilnejših tipov sistemov za podporo upravljanja s konfiguracijami izdelkov in njihove PO, t.j. PDM in SCM sistemov:

- *PDM sistemi* so se začeli množičneje uporabljati v osemdesetih letih prejšnjega stoletja. Prvi PDM sistemi so bili usmerjeni predvsem v podporo inženirskemu procesu (posebej načrtovanju) in proizvodnji. Ker je ta dva procesa dokaj lahko določiti, so PDM sistemi v začetku vsiljevali stroge inženirske postopke, kar se je tedaj, vsaj kar se tiče industrijske proizvodnje, pokazalo kot dobro. Iz tega časa v največji meri izhaja tipična zgradba PDM sistema. V devetdesetih letih se je usmerjenost na izboljšave preselila tudi na druga poslovna področja, funkcije in procese (nabava, servisne storitve oziroma podpora izdelkov ipd.). To je organizacijam pomagalo časovno uskladiti ter voditi vse aktivnosti v zvezi z izdelki skozi njihov celotni življenjski cikel. Skrajšati čas prihoda na tržišče (angl. Time-to-market) je bila ena najpomembnejših nalog za večino podjetij. V zadnjem času

se za PDM razvijajo vse bolj trdožive zgradbe (robustne arhitekture) v Web okolju. S pomočjo te tehnologije, ki jo v zadnjih letih vse bolj uporabljajo v sistemih za upravljanje življenjskega cikla izdelkov, so nastale še večje možnosti za izboljšave in pohitritve procesa obdelave informacij o proizvodih (Aberdeen Group, 2001, str. 2). To vodi do dodatnega krajšanja časa, potrebnega za snovanje, razvoj in prihod izdelkov na tržišče. Vsaj za nekatere funkcije PDM sistema, kot npr. prijava napak na izdelkih (angl. helpdesk), vse bolj do izraza prihajajo orodja in aplikacije za upravljanje odnosov s kupci (angl. Customer Relationships Management, krat. CRM), ki se lahko kot vmesnik do kupcev navezujejo na PDM sisteme. Novejši PDM sistemi poskušajo ponuditi funkcije za vse bolj celovit način upravljanja s proizvodi, poleg zgoraj naštetih poslovnih področij vse bolj podpirajo tudi razvoj in obvladovanje PO, upravljanje s projekti, logistiko itd. Vendar niti novejši PDM sistemi (Taramaa, 1998, str. 81) še niso v celoti primerni za upravljanje konfiguracij PO.

- Medtem ko so v začetnih fazah uporabe *SCM sistemov* na tržišču prevladovali sistemi, ki so obvladovali in uporabljali le navadne datoteke, so bili ti pozneje nadgrajeni z odlagališči podatkov oziroma projektov, ki so vsebovali tudi (običajno relacijsko) podatkovno bazo. Novejša orodja gredo vse bolj proti uporabi transparentnih datotek, podpori geografsko porazdeljenemu razvoju, podpori upravljanju nalog, obvladovanju komponent, vse bolj pa je opazen tudi pomik proti uporabi odjemalcev za Web okolje. SCM sistemi so po mnenju mnogih strokovnjakov s tega področja dosegli precejšnjo mero zrelosti. Danes so v uporabi in na tržišču dostopna zelo učinkovita orodja s širokim obsegom funkcionalnosti in sposobnostjo učinkovite avtomatizacije procesa.

Priloga E - Procesni pristop RUP (Rational Unified Process)

Rational Unified Process (krat. RUP) je proces razvoja programske opreme, ki opredeljuje vlogo in naloge posameznikov in razvojne skupine kot celote (Roubine, 2001, str. 22). RUP predstavlja "ogrodje objektnega procesnega modela", ki ga je možno prilagoditi potrebam različnih organizacij. Ustrezno prilagoditev procesa zahtevam dane organizacije dosežemo z uporabo orodja Rational Process Workbench. To orodje deluje v tesni povezavi z aplikacijo Rational Rose, ki se uporablja za modeliranje zahtev uporabnika. Splošne lastnosti RUP:

- RUP je proces v programskem inženirstvu in se usmerja v razvoj in vzdrževanje modelov, t.j. semantično bogatih predstavitev razvojnega sistema;
- vsebuje vodila, kako učinkovito uporabljati UML. Ta je bil prvotno zasnovan s strani Rationalovih strokovnjakov in je pozneje postal industrijski standard, trenutno pa ga vzdržuje organizacija OMG (Object Management Group);
- podprt je z množico orodij za avtomatizacijo procesa programskega inženirstva;
- je nastavljiv in prilagodljiv proces.

V primerjavi s tradicionalnim slapovnim (angl. waterfall) modelom razvojnega življenjskega cikla RUP podpira ciklični, iterativni način razvoja (glej sliko 36). Ta je običajno precej bližje realnosti kot linearni način razvoja, katerega podpirajo tradicionalni modeli.

Slika 36: Pristop k upravljanju življenjskega cikla proizvoda z RUP

Vir: Probasco, 2000, str. 5

V zasnovo RUP je vgrajenih šest v praksi preverjenih in potrjenih najboljših izkušenj pri razvoju PO (Roubine, 2001, str. 23). Dobljene so bile na podlagi spremljanja razvoja PO pri Rationalovih strankah. Opisane so v nadaljevanju.

1. *Razvoj po korakih (iterativni razvoj).* Za razliko od slapovnega modela, kjer življenjski cikel sledi dobro znanemu zaporedju faz, RUP življenjski cikel sistemov razdeli na več ciklov. V prvem ciklu je proizvod izdelan, vsak naslednji cikel pa se ukvarja z novo generacijo izdelka. RUP razdeli razvojni cikel na naslednje štiri faze:
   a) *začetna faza (angl. inception phase).* Glavni cilji te faze so: doseči sodelovanje na projektu med izdelovalcem PO in vsemi ključnimi uporabniki; določitev vsebine projekta in omejevalnih pogojev; ocena celotnih stroškov in časovnega obsega projekta; ocena izvorov nepredvidljivih dogajanj in kritičnih primerov uporabe sistema;

b) *faza zbiranja informacij (angl. elaboration phase).* Namen te faze je: analiza problemske domene; vzpostavitev kakovostne zgradbe (arhitekture) sistema; izdelava projektnega plana in odstranitev največjih tveganj. Pri tem poleg zgradbe in vizije sistema nastane podroben načrt za naslednjo fazo (faza izdelave);
c) *faza izdelave (angl. construction phase).* V tej fazi se osredotočimo na podrobnejše načrtovanje, vgradnjo in testiranje. Izdelajo se vse komponente sistema in se vgradijo v končni izdelek, pri čemer se vse lastnosti predhodno temeljito preverijo. Glavni cilji: zmanjševanje stroškov razvoja na račun optimalnega upravljanja z viri; doseganje ustrezne kakovosti; izdelava uporabnih različic (alfa, beta ipd.);
d) *prevzemna faza (angl. transition phase).* Namen te faze je uspešna predaja izdelka končnim uporabnikom. Nekatere od ključnih aktivnosti te faze so izdelava plana namestitve PO, izvedba namestitve pri stranki in izobraževanje uporabnikov.

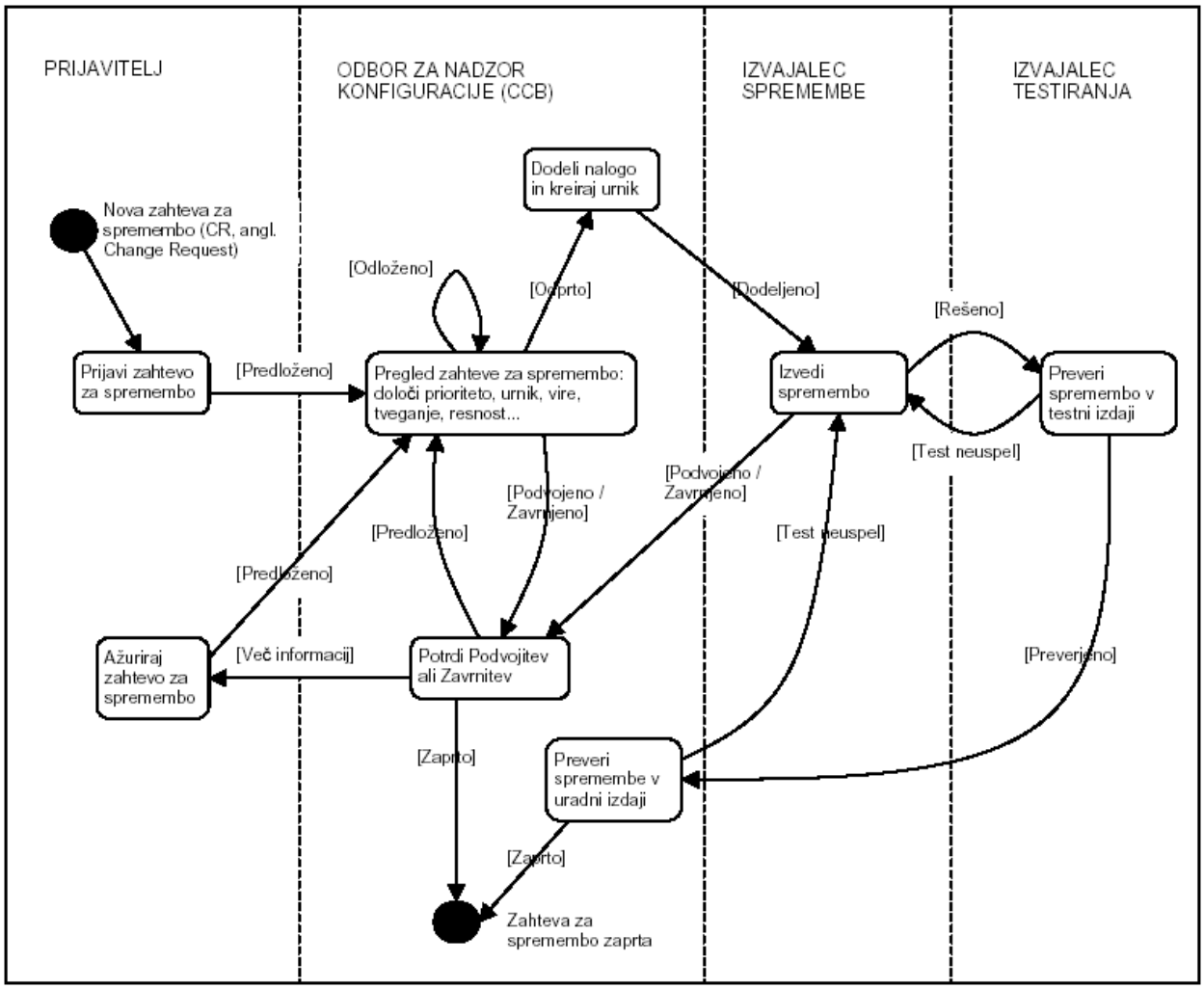
Razen posebnega primera, ko se razvojna pot izdelka ustavi že po začetnem ciklu, je izdelek deležen razvoja z večkratno ponovitvijo vseh štirih faz. Vsaka faza se zaključi z dobro določenim mejnikom, t.j. s časovno točko, kjer morajo biti opravljene kritične odločitve in doseženi ključni cilji.

2. *Upravljanje z zahtevami.* Upravljanje z zahtevami je proces sistematičnega zbiranja, organiziranja in dokumentiranja zahtev za nek kompleksen sistem. Poteka skozi celoten življenjski cikel sistema. RUP obravnava ta proces skozi šest faz (Roubine, 2001, str. 26):
   a) *analiza problema* - proces razumevanja problema in potreb uporabnika, vključno z izdelavo predloga, ki ustreza potrebam uporabnika. Tu so intenzivno zajete tehnike modeliranja poslovnih procesov, sistemskega inženiringa ipd.;
   b) *razumevanje potreb uporabnika* - tu so zajeti prijemi in tehnike z namenom čim bolje razumeti uporabnika. Primeri: pogovori z uporabnikom, uporaba visokonivojskih opisov sistema, delavnice na temo zbiranja zahtev itd.;
   c) *definiranje sistema* - v tej fazi gre za prehod k določanju ustrezne rešitve, k t.im. *rešitveni domeni*. Za kompleksne sisteme je potrebno uporabiti natančne strategije in tehnike za obdelavo zahtev;
   d) *upravljanje z obsegom* - večina projektov se začne s precej večjim obsegom zahtev za proizvod, kot jih je mogoče v načrtovanem obsegu projekta uresničiti. V tej fazi se določijo osnovne zahteve za izdelek, določijo se nivoji težavnosti in tveganja za vsako lastnost, določijo se načini sodelovanja s stranko, izdelajo se natančni modeli v zvezi s tem, kdo bo kaj, kdaj in kako delal;
   e) *podrobnejša definicija sistema* - opredelitev vhodnih in izhodnih parametrov ter atributov za sistem in njegovo okolje;
   f) *izdelava pravega sistema* - načrtovanje in izvedba je najzahtevnejša naloga. Pri tem se uporablja vrsta metod in tehnik, npr. primeri uporabe (use case) za načrtovanje, analitični postopki preverjanja pri spremljanju razvoja izdelka in projekta, s hkratno uporabo tehnik sledljivosti in povezovanja z drugimi, sorodnimi projekti. V tej fazi je potrebno uporabljati tudi tehnike, vezane na nadzor sprememb na izdelkih.
3. *Uporaba komponentne zgradbe sistema.* Zgradba sistema je zelo pomembna za upravljanje različnih pogledov na sistem, prav tako pa tudi za nadzor celotnega življenjskega cikla razvoja PO. Ne zajema le strukture in vedenja sistema, pač pa tudi uporabo, funkcionalnost, učinkovitost, prožnost, možnost ponovne uporabe posameznih delov sistema, razumljivost, ekonomičnost in tehnološko omejenost.
4. *Vizualno modeliranje.* Model je poenostavitev resničnosti, ki v celoti opisuje sistem z določene perspektive. En ključnih namenov modeliranja je boljše spoznati in razumeti sistem. Modeliranje je pomembno, ker pripomore k višjemu, kakovostnejšemu upravljanju

zgradbe sistema. Pri kompleksnejših sistemih to pomeni bistveno boljše upravljanje s temi sistemi. Uporaba UML omogoča učinkovito komuniciranje in sprejemanje kakovostnih odločitev.

5. *Pogosto preverjanje kakovosti PO.* Zelo pomembno je nenehno ocenjevanje kakovosti sistema. Pri tem se daje poudarek posameznim ključnim parametrom izdelkov (funkcionalnost, zanesljivost, učinkovitost...).
6. *Nadzor nad spremembami.* En največjih izzivov pri razvoju PO je obvladovanje aktivnosti in izdelkov razvijalcev, organiziranih v razvojne skupine, ki lahko delajo na različnih lokacijah, na večih platformah, na večih izdelkih in projektih, v različnih fazah življenjskega cikla.

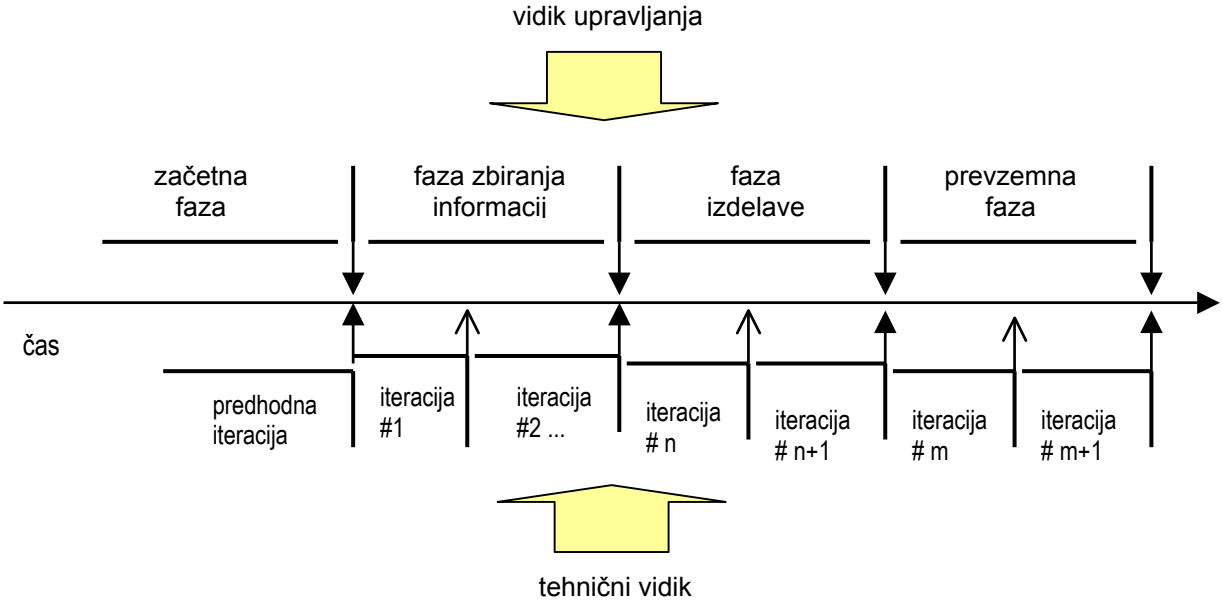
Slika 37: Proces obdelave zahtev za spremembo, kot ga predvideva RUP



Vir: Marand, 2001, str. 16

Pri tem se RUP usmerja v podporo tako upravljalskega kot tehničnega vidika procesa (slika 38).

Slika 38: Upravljalski in tehnični vidik procesa



Vir: Kruchten, 1996, str. 3

RUP ni usmerjen v dokumentacijo, temveč v sam programski izdelek. Pri tem naj bi dokumentacija ne bila preobsežna. V procesu naj bi nastopali le tisti dokumenti, ki dejansko dodajajo vrednost projektu, ali z vidika vodenja ali s tehničnega vidika. Priporoča se naslednja množica dokumentov (Kruchten, 1996, str. 11):

1. *dokumenti z zvezi z upravljanjem oziroma vodenjem projekta* ne sestavljajo proizvoda. Uporabljajo se za vodenje ali spremljanje poteka projekta, ocenjevanje tveganj, upravljanje z viri, informiranje naročnika o poteku projekta. To so: dokument z opisom organizacijske politike - opis instance tega generičnega procesa; dokument z zahtevami in prioritetami na nivoju celotnega sistema; dokument s poslovnim primerom, ki vsebuje pogodbeni in finančni kontekst, vključno z oceno finančnih kazalnikov, (npr. ROI); razvojni plan, vsebujoč tako celotni razvoj kot trenutno in naslednjo iteracijo, cikel; dokument s kriteriji za ocenjevanje, ki vsebuje kriterije sprejemljivosti in druge tehnične cilje za posamezne glavne faze projekta (ta dokument nastane med dvema glavnima mejnikoma v projektu); opis posamezne izdaje proizvoda oziroma sistema; dokument z opisom procesa dostavljanja in razširjanja proizvoda po terenu, z opisom procesov dostave, šolanja uporabnikov, nameščanja, prodaje, proizvodnje itd.; dokument z oceno stanja projekta, ki se izdeluje periodično in meri napredek projekta, njegove rezultate, stroške, kritična tveganja ipd.;
2. *tehnična dokumentacija* lahko vsebuje tako sam sistem (izvedbeno kodo, priročnike...) kot dokumente, ki se uporabljajo za proizvodnjo tega sistema (modeli PO, izvorna koda in druge informacije, potrebne za nastanek izdelka). To so: uporabniška dokumentacija, razvita zgodaj v življenjskem ciklu izdelka; dokumentacija z opisom PO, npr. uporabniški primeri, diagrami procesov in drugi modeli, izdelani in vzdrževani s pomočjo CASE orodij; dokument z opisom zgradbe sistema, ki nastane s pomočjo dokumentacije, v kateri je opisana PO, in vsebuje opise strukture programskega izdelka, njegove razgradnje (v podsisteme, procese itd.), opise glavnih vmesnikov in ključne dejavnike, na podlagi katerih so bile sprejete odločitve v zvezi z zgradbo sistema.

Priloga F - Rational ClearQuest

Rational ClearQuest je aplikacija, namenjena sledenju in upravljanju zahtev za spremembo proizvodov (Marand, 2001, str. 9). Zahteve za spremembo se shranjujejo kot zapisi v ClearQuestovi bazi podatkov. ClearQuest podpira različne tipe zapisov, npr. tip za napake na proizvodu ali tip za aktivnost. Pri tem ima vsak tip zapisa posebne zahteve za svoja polja. T.im. *shema* se nanaša na vse atribute, ki določajo podatkovno bazo orodja ClearQuest. ClearCase ima vnaprej določene sheme, prav tako pa omogoča njihovo prilagajanje oziroma ustvarjanje novih shem. ClearQuestovi zapisi se vodijo skozi svoj življenjski cikel. Vsaka stopnja v življenjskem ciklu se imenuje *stanje* (angl. state), vsakemu prehodu iz enega stanja v drugo pa pravimo *prehod med stanji* (angl. state transition). ClearQuest sestavljajo tri ločene komponente:

- *ClearQuest* - uporablja se za predložitev, spreminjanje in sledenje zahtev za spremembo; poleg tega je namenjen spremljanju poteka projekta z izvajanjem poizvedb, grafičnimi predstavitvami podatkov o poteku projekta in poročili;
- *ClearQuest Designer* - namenjen je administratorjem orodja ClearQuest za prilagoditev tega orodja CRM (angl. Customer Relationships Management, upravljanje odnosov s kupci) procesom, upravljanju ClearQuestove baze podatkov in administriranju uporabnikov oziroma uporabniških skupin.
- *ClearQuest Web* - namenjen je predložitvi zahtev za spremembo, izvajanju poizvedb in doseganju ClearQuestove podatkovne baze, z uporabo spletnih brskalnikov.

Glavni namen ClearQuesta (Marand, 2001, str. 10) je na enem mestu zbrati in voditi informacije, povezane z aktivnostmi v zvezi s spremembami v projektu. Pri tem:

- *razvojni inženirji* lahko identificirajo in določajo prioritete akcijam, ki se nanašajo na njihov del kode;
- *inženirji, ki izvajajo testiranje*, lahko nadzirajo tekoče stanje v življenjskem ciklu reševanja zahteve za spremembo, z namenom preverjanja kakovosti PO;
- *vodje projektov* lahko dobe ažurne informacije, ki jim omogočajo razporejanje projektnih virov, usmeritev pretoka podatkov in določitev rokov za izdajanje programske opreme;
- *administratorji* lahko integrirajo ClearQuest z obstoječimi orodji in ga prilagodijo, tako da ustreza obstoječim postopkom.

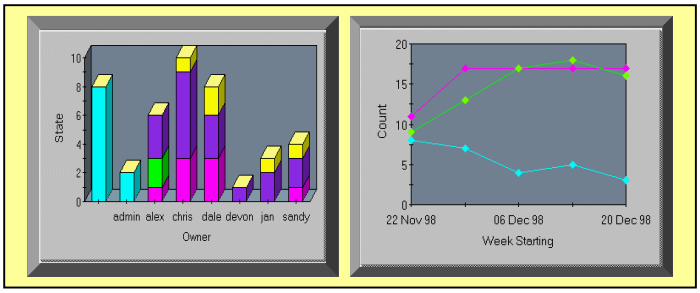
V kratkem si oglejmo nekaj najpomembnejših značilnosti procesa, ki ga podpira ClearQuest (Marand, 2001, str. 9):

- *predložitev zahteve za spremembo.* Uporaba ClearQuesta se začne s predložitvijo zahteve za spremembo. S tem se ustvari zapis v bazi podatkov, ki ga lahko spremlja vsak član razvojne skupine, tako v fazi razvoja kot v fazi testiranja. Lahko se določijo različni tipi zahtev za spremembo;
- *delo z zapisi o zahtevah za spremembo.* Uporabnik dela z zahtevami za spremembo tako, da jih vodi skozi različne stopnje oziroma stanja. V vsakem stanju lahko uporabnik izvede različne akcije, npr. spremembo zapisa ali pomik na naslednje stanje. T.im. *akcijski menu* poda seznam možnih akcij na zapisu, ki se nahaja v danem stanju. Primer: zapis je lahko na začetku v stanju "predložen" (angl. "submitted"), nato "dodeljen" (angl. "assigned") komur koli od članov projektne skupine; lahko je "odprt" (angl. "opened"), ko kdor koli dela na njem, pozneje pa rešen (angl. "resolved"),

"ovrednoten" (angl "validated") in na koncu "zaprt" (angl. "closed"). Dovoljena stanja za dani tip zapisa in prehodi med njimi se lahko določijo poljubno, na ravni projekta;

- *uporaba poizvedb.* Poizvedbe predstavljajo način dela s ClearQuestovo bazo podatkov. Uporabniki običajno s poizvedbami iščejo zapise, ki jih nameravajo obdelati. Primer: uporabnika lahko zanimajo npr. zapisi, ki zadevajo le njega ali le zapise, ki so povezani z njegovim projektom. Poizvedbe temeljijo na t.im. *filtrih*. V kolikor uporabnik ne uporabi nobenega filtra, se izpiše seznam vseh zapisov iz podatkovne baze. Rezultati so podani v tabelarični obliki. Uporabnik jih lahko preuredi in oblikuje ustrezno poročilo;
- *analiza podatkov s pomočjo grafičnih predstavitev.* ClearQuest premore tri različne tipe diagramov, ki so uporabniku v pomoč pri analiziranju podatkov:
  (a) *diagrami za predstavitev porazdelitev* se uporabljajo za predstavitve različnih kategorij in vrednosti. Primera uporabe: ugotavljanje, komu je bilo dodeljenih v reševanje največ zahtev za spremembo ali kateri zapisi imajo najvišjo prioriteto;
  (b) *diagrami za predstavitev tendenc* prikazujejo, za koliko zapisov je bil izveden prehod v določeno stanje na določen dan, teden ali mesec. Prikazujejo razmerja in hitrost, s katero so nove zahteve za spremembe predložene, rešene ali promovirane v naslednja stanja;
  (c) *časovni diagrami* omogočajo sledenje zgodovine prehodov med stanji zapisov. Ta vrsta diagramov se lahko npr. uporablja za odgovore na vprašanje tipa: "Koliko problemov je za dani proizvod odprtih en (dva, tri...) teden?";
- *prilagajanje.* Z orodjem ClearQuest Designer lahko administrator izvaja naslednje operacije: (a) izdelovanje in spreminjanje shem, vključujoč spreminjanje form, namenjenih predložitvi in spreminjanju zapisov; (b) ustvarjanje in spreminjanje podatkovnih baz; (c) dodeljevanje in spreminjanje dovoljenj za uporabnike in skupine itd. Za prilagajanje prehodov med stanji zapisov se uporablja t.im. *matrika prehodov med stanji* (angl. state transition matrix);
- *izdelava poročil.* Uporabnik lahko izvaja različne poizvedbe nad podatki o zahtevah za spremembo in jih z raznimi pripomočki oblikuje v raznovrstna grafična poročila.

Slika 39: Primerka grafičnih poročil v ClearQuestu



Vir: Interna dokumentacija podjetja (korespondenca s podjetjem Rational)

Priloga G - Najboljše izkušnje ali pravila dobre prakse na področju upravljanja konfiguracij izdelkov

Vsako podjetje ima edinstvene proizvode, ki zahtevajo lastno izvedbo procesa upravljanja konfiguracij. Najboljše izkušnje ali pravila dobre prakse (angl. best practices) podjetjem omogočajo, da v svoje sisteme vgradijo procesne pristope, ki so se dobro obnesli v mnogih drugih okoljih. Proces upravljanja konfiguracij v danem poslovnem sistemu se tako lahko uredi kot zbirka najboljših izkušenj oziroma preverjenih in široko sprejetih ter uporabljenih načinov dela. Te je vsakič potrebno ustrezno prilagoditi pogodbi, projektu in izdelku, v zvezi s katerim se uporabljalo (Lyon, 2001, str. 112). Posebej pri novih razvojnih programih ali takšnih, ki se od prejšnjih bistveno razlikujejo, je koristno v proces upravljanja konfiguracij izdelkov vključiti čim več najboljših izkušenj s tega področja (Lyon, 2001, str. 23). Za proces upravljanja konfiguracij so zelo značilna naslednja pravila dobre prakse (Lyon, 2001, str. 122):

- *Usklajevanje zahtev za upravljanje konfiguracij.* Način upravljanja konfiguracij je potrebno uskladiti potrebam poslovnega procesa, v katerem bo izdelek razvit, po drugi strani pa tudi glavnim željam naročnika. To se izvede s pogajanji, pri čemer so na eni strani zahteve poslovnega sistema, na drugi pa interesi kupca. Pomembno je, da so pri tem v celoti upoštevani interesi funkcionalnih organizacijskih enot podjetja, ki pri pogajanjih aktivno sodelujejo. Ko so interesi kupca in poslovnega sistema glede načina upravljanja konfiguracij usklajeni, se rezultati pogajanj dokumentirajo v planu upravljanja konfiguracij (glej naslednjo alinejo). S temi pogajanji, katerih rezultat je ustrezna dokumentacija z zahtevami za način upravljanja konfiguracij izdelka, se zagotovi temelj za ustrezno komunikacijo s kupcem v toku izvedbe projekta. Dve od pomembnih pričakovanih prednosti v odnosu "kupec-proizvajalec" (oziroma "naročnik-dobavitelj") pri tem sta: (a) dogovorjen način upravljanja konfiguracij zagotavlja lažje sodelovanje in bolj usklajeno komunikacijo med naročnikom in pogodbenim partnerjem v toku projekta; (b) planiranje in izvedba presoj konfiguracije zagotavlja skladnost dobavljenega izdelka z zahtevami iz pogodbe.
- *Planiranje upravljanja konfiguracij za projektni program.* Plan upravljanja konfiguracij je poleg strategije upravljanja konfiguracij in sistema orodij za podporo upravljanja konfiguracij en od treh ključnih dejavnikov uspešne uporabe upravljanja konfiguracij (Bounds, 1993, str. 6). Planiranje upravljanja konfiguracij je v tesni povezavi z že omenjeno aktivnostjo pogajanja z naročnikom glede zahtev za upravljanje konfiguracij. Plan upravljanja konfiguracij za projekt ali za projektni program izdela skupina inženirjev, odgovorna za upravljanje konfiguracij v omenjenem projektu (oziroma projektnem programu). Pri planiranju se dokumentirajo roki in način izvedbe ustreznih aktivnosti v projektu, ki bodo zagotovile, da bodo projekt in njegovi rezultati ustrezali postavljenim ciljem. Pomembno je, da se določijo naloge in aktivnosti, povezane s presojami rezultatov načrtovanja. Plan vključuje ustrezen urnik izvedbe, ki naj bi bil zaključen pred prvim formalnim pregledom rezultatov načrtovanja. V urniku so določeni roki za izvedbo prehodov na višje nivoje nadzora, formalne presoje, zajemanje planiranih izhodiščnih konfiguracij izdelkov ipd. V plan naj bi bil vključen tudi seznam vseh vhodov v proces upravljanja konfiguracij. Ta seznam naj bi se preveril ob izvedbi prvega formalnega pregleda rezultatov načrtovanja. V praksi se pogosto dogaja, da so izboljšave na področju upravljanja konfiguracij le del širšega prizadevanja za izboljševanje poslovnih

procesov. To pomeni, da mora biti v takšnem primeru plan upravljanja konfiguracij usklajen z vsemi drugimi plani, ki se nanašajo na izboljšave.

- *Uporaba izhodiščnih konfiguracij.* Izhodiščna konfiguracija je strukturirana množica vseh gradnikov (t.j. dokumentov v zvezi z načrtovanjem, datotek z izvorno kodo itd.), ki jih vodimo v življenjskem ciklu danega izdelka in ki predstavljajo ta izdelek v dani časovni točki. Pomembna ideja v zvezi z izhodiščnimi konfiguracijami je ta, da najprej identificiramo seznam novih ali spremenjenih gradnikov. Ko jih zajamemo v vnaprej izbrani časovni točki, dobimo prikaz vseh podatkov o izdelku v danem trenutku. Od tu naprej se predlagane spremembe na tem izdelku vodijo bolj nadzorovano, na bolj formalen način kot do tedaj. Spremembe se nato praviloma izvajajo le v primeru, ko predloge za spremembo odobre t.im. odbori za nadzor konfiguracije (angl. Configuration Control Boards, krat. CCB). Razvoj gre v smeri avtomatiziranja zajemanja gradnikov pred izdelavo izhodiščnih konfiguracij, v katerih gradniki nastopajo. To pa je pogojeno tudi z izboljšanjem možnosti povezovanja PDM sistemov z drugimi sistemi in okolji, ki se v procesu načrtovanja in realizacije izdelkov uporabljajo. V preteklosti so se v večini projektov pojavljali naslednji osnovni tipi izhodiščnih konfiguracij (Angstadt, 2000, str. 26): *funkcionalna* - vsebuje strukturo zahtev; *dodeljena* - vsebuje rezultate načrtovanja; *proizvodna* - vsebuje končno strukturo izdelka, proizvoda. Te vrste izhodiščnih konfiguracij izgubljajo pomen v okoljih, kjer vse bolj prevladuje programiranje spletnih aplikacij. V tovrstnih okoljih obstaja težnja (Angstadt, 2000, str. 26), da se namesto izhodiščnih konfiguracij uporabljajo trenutno objavljene verzije gradnikov (izvorna koda, dokumenti, zahteve...). Vsak odbor za nadzor sprememb (CCB) naj bi v ustreznem trenutku življenjskega cikla izdelka zagotovil izdelavo in nadzor tiste vrste izhodiščnih konfiguracij, za katere je pooblaščen (Sorensen, 1999, str. 5).
- *Delovanje odbora za nadzor konfiguracij.* Odbori za nadzor konfiguracij (angl. Configuration Control Board, krat. CCB) pridejo do veljave pri prenosu gradnikov (npr. dokumentov) iz razvijalčevega delovnega okolja v uradno okolje oziroma pod nadzor sistema za upravljanje konfiguracij. Sestave odborov za nadzor konfiguracij so v praksi odvisne od narave projektov ali projektnih programov. Odbori so lahko interni (sestavljeni le iz zaposlenih v danem podjetju) ali eksterni (vključeni npr. predstavniki naročnikov proizvodov). Člane običajno izbere vodja projektnega programa ali nekdo z enakovrednimi pooblastili. V odboru naj bi bili vsaj predstavniki razvijalcev oziroma načrtovalcev sistemov, proizvodnje, zagotavljanja kakovosti, upravljanja konfiguracij in servisne podpore na terenu. Glavni namen teh odborov je pregledati vpliv predlaganih sprememb na zasnovo oziroma načrt proizvoda. Pri tem se pregleda tehnični vpliv spremembe na zasnovo proizvoda, ocenijo se vplivi spremembe na proizvodnjo, vplivi na stroške in izvajanje plana projekta ipd. S tem pristopom si odgovorni za zagotovitev končnega proizvoda pridobe dovolj informacij, ki jim omogočajo sprejemanje ustreznih poslovnih odločitev.
- *Identificiranje in obravnavanje problemov.* Prvi korak v procesu nadzora sprememb je identificiranje in reševanje sprememb. Na probleme opozorijo nosilci ključnih vlog v procesu: načrtovalci, izvajalci gradnje programskih paketov, verifikatorji, vzdrževalci, lahko pa tudi kupci oziroma uporabniki na terenu. Stopnja formalnosti postopka obravnave problemov je lahko različna. Ta se lahko izvaja s sestanki odbora za razrešitev problemov (angl. Problem Resolution Board, krat. PRB), s sestanki komitejev za korektivne ukrepe (angl. Corrective Action Committee, krat. CAC) ali z neformalnim stikom s projektnimi vodji, ključnimi razvijalci ipd. Pri tem je le pomembno, da problem dobi ustrezno pozornost s strani ustreznih ljudi. Reševanje določenega problema lahko

zahteva spremembe v zasnovi oziroma zgradbi izdelka. V takšnih primerih odbor za reševanje problemov na to opozori odbor za nadzor konfiguracij. Odbor za razrešitev problemov ni zadolžen za obravnavanje in potrjevanje sprememb v zgradbi (arhitekturi) izdelka, le predlaga jih odboru za nadzor konfiguracij.

- *Delovanje odbora za razrešitev problemov (angl. Problem Resolution Board, krat. PRB).* Namen PRB je identificirati in razrešiti probleme, ki se pojavijo v fazah proizvodnje in vzdrževanja. Naloga PRB je sprejeti, oceniti (ovrednotiti), raziskati in posredovati identificirane probleme. Pri tem PRB: določi naloge, potrebne za rešitev problemov; neposredno reši probleme, v kolikor je to mogoče; dokumentira vhode in določi stanja in postopke, potrebne za zagotovitev sledljivosti reševanja problemov; poroča vodstvu in kupcem o stanju procesa reševanja problemov, kot je predvideno z internimi postopki in kot je dokumentirano v planu upravljanja konfiguracij. PRB poskrbi, da so vsi njemu poverjeni problemi učinkovito in pravočasno rešeni. Prav tako poskrbi za natančno določitev tehničnih, stroškovnih in časovnih vplivov na dani projektni program. Vloge in naloge članov v odboru: *predsednik* zagotoviti, da se procesi, povezani s tem odborom, izvajajo v skladu s plani; *administrator* je odgovoren za administriranje podatkov v zvezi s problemi in sestanki; *projektni inženir* pregleda vpliv problema na zasnovo izdelka, oceni vpliv na stroške in časovni plan projekta; *predstavniki funkcijskih organizacijskih enot* so odgovorni za usklajevanje glede stanj gradnikov, ki jim jih priredi PRB.
- *Predlaganje, obravnava in izvajanje sprememb.* V zvezi s predlaganjem, obravnavo (potrjevanjem/zavračanjem) in izvajanjem sprememb, tako v načrtu izdelka kot na samem izdelku, je potrebno določiti, postaviti in vzdrževati formalen proces. Ta mora ustrezati zahtevam poslovanja in naravi izdelka. Pri tem naj bi bil proces čim bolj avtomatiziran, brez nepotrebne uporabe papirja. Proces potrjevanja sprememb se običajno začne s predlogom za izvedbo spremembe, ki je predložen odboru za nadzor konfiguracije. Ta odbor nato obravnava vpliv spremembe na zgradbo in konfiguracijo izdelka. Odbor se lahko sestaja na formalnih sestankih ali pa proces odločanja izvaja na korespondenčen način, s pomočjo temu namenjenega podpornega sistema. V kolikor se predlog spremembe zavrne, se v sistemu za vodenje podatkov o izdelkih zabeleži ustrezen zapis o tem, sprememba pa se ne izvede. V nasprotnem primeru, če je predlog spremembe potrjen, gre sprememba v naslednjo fazo procesa. Spremembe se izvedejo tako na zasnovi proizvoda kot na samem izdelku.
- *Zajemanje, obdelava in poročanje o stanjih v zvezi s konfiguracijami.* Vodenje stanj konfiguracij (angl. configuration status accounting, krat. CSA) je podproces procesa upravljanja konfiguracij, namenjen spremljanju informacij o trenutnem stanju potrjenih osnovnic ali izhodiščnih konfiguracij izdelkov in spremljanju napredka in stanj predlaganih in odobrenih sprememb načrtovanja. Pri tem se lahko vodijo tudi razna druga stanja, npr. stanja v zvezi z neskladnostmi, ki se lahko pojavijo v konfiguracijah izdelkov ali njihovih komponent. Aktivnosti vodenja stanj konfiguracij lahko razdelimo v tri različna, a prekrivajoča se področja:
  (a) *zajemanje podatkov.* Obstaja več tipov podatkov, ki se pojavljajo in jih potrebujemo v procesu upravljanja konfiguracij izdelkov. V grobem jih lahko razdelimo na dva tipa: (i) podatki, ki jih je nujno potrebno upoštevati in so zapisani v pogodbi; (ii) podatki, ki so koristni in potrebni za nemoteno spremljanje, nadzorovanje in izvajanje aktivnosti v zvezi s proizvodi. Podatki se lahko beležijo na različne načine, npr. v papirnati ali digitalni obliki, lahko se vnašajo ročno ali samodejno, npr. s pomočjo sprožilcev ob določenih nadzorovanih in izvedenih dogodkih;

(b) *obdelava podatkov* vsebuje spreminjanje podatkov v obliko, lahko služi kot izhodišče za nadaljnje aktivnosti. Na podlagi teh podatkov se lahko npr. izdelajo poročila, namenjena naročnikom izdelkov, z namenom obvestiti jih, da pogodbene aktivnosti potekajo v skladu s planom. Podatki so lahko namenjeni tudi izdelavi različnih analiz, npr. v zvezi z ugotavljanjem in optimizacijo trajanja različnih ciklov v procesu;

(c) *poročanje o stanjih konfiguracije* je aktivnost, pri kateri se izdelajo in dostavijo poročila o informacijah, ki so trenutno na voljo v sistemu. Lahko se izdelujejo različni tipi poročil, v različnih oblikah, za različne naročnike.

- *Uporaba presoj konfiguracij.* Obstojajo trije tipi presojanja konfiguracij, ki se dokaj strogo izvajajo v ameriški obrambni industriji, pa tudi v raznih poslovnih sistemih:

  (a) *presoja funkcijske konfiguracije* (angl. functional configuration audit, krat. FCA) je namenjena potrditvi, da je bila na sistemskem nivoju pri načrtovanju ustrezno izpolnjena vsaka zahteva iz specifikacije. V kolikor določene zahteve ni mogoče praktično preveriti, se izdela teoretična ocena ustreznosti. Na omenjene teste se običajno sklicujemo s pojmi, kot sta ocenjevanje načrtovanja in kvalifikacijski testi;

  (b) *presoja fizične konfiguracije* (angl. physical configuration audit, krat. PCA) se izvede po uspešno izvedeni presoji funkcijske konfiguracije. Med presojo fizične konfiguracije se izvede neposredna primerjava med načrtovalnimi podatki ali načrtovalnimi risbami in fizičnimi lastnostmi proizvedenih kosov ali izdelkov. Preverjajo se tako različne mere kot drugi tehnični podatki, zajeti v načrtovanju teh delov proizvoda. Ker se preverjanje navadno izvede na prvem proizvedenem primerku sestavnega dela izdelka, so ga v preteklosti imenovali tudi *pregled prvega člena* (angl. first article inspection, krat. FAI). Med presojo fizične konfiguracije se izvede tudi pregled skladnosti osnovnic ali izhodiščnih konfiguracij (npr. tipa "as-defined") z dejanskim stanjem. Končna aktivnost pri presoji fizične konfiguracije je izvedba *testa sprejemljivosti* (angl. acceptance test, krat. AT), kjer se preveri oziroma zagotovi, da je dobljeni del proizvoda v fizičnem in funkcionalnem pogledu enakovreden tistemu, ki je prestal presojo funkcionalne konfiguracije. Test sprejemljivosti je običajno sestavljen iz podmnožice ocenjevanj načrtovanja in kvalifikacijskih testov, uporabljenih pri presojah funkcijskih konfiguracij;

  (c) *presoja z namenom preverjanja konfiguracije* (angl. configuration verification audit, krat. CVA) se izvede na vsakem proizvedenem delu proizvoda, ki je namenjen dobavi naročniku. Pri tem se preverijo skladnosti v zvezi z revizijami oziroma verzijami gradnikov, njihovimi identifikacijskimi številkami ipd. Seznami, ki se uporabljajo v ta namen, se izdelajo že ob začetku razvojne faze izdelka. Ti seznami vsebujejo gradnike, ki jih je potrebno pregledati. Pri tem je pomembno, da se pregled usmeri v tiste gradnike, pri katerih je pregled dejansko potreben. Po eni strani je potrebno zagotoviti sledljivost, po drugi upoštevati stroške za pregled. Pri odločanju glede gradnikov, ki naj bodo vključeni v seznam oziroma pregledani, je potrebno vzeti v obzir še tehnično tveganje (npr. v primeru načrtovanja novih sestavnih delov ali izdelkov), zanesljivost dobaviteljev ipd.

Priloga H - Projekti uvajanja sistemov za upravljanje konfiguracij izdelkov

V nadaljevanju je v kratkem predstavljena problematika v zvezi s projekti uvajanja CM sistemov. Ti sistemi se v podjetja običajno uvajajo s pomočjo posebnih, temu namenjenih projektov. Tako kot pri vseh projektih, se tudi za te izdelajo projektni plani, izvajanje projekta pa poteka po fazah. Na ta način se zmanjša tveganje pri prehodu na nov sistem, saj se manjše spremembe lažje obvladujejo. Efstathiou (Efstathiou, 2002, str. 2) loči naslednje tipe projektov, povezanih z uvajanjem novih informacijskih tehnologij:

- projekti z namenom povečanja učinkovitosti določenega poslovnega procesa;
- projekti z namenom nižanja stroškov proizvodov (vključno z njihovim razvojem in vzdrževanjem);
- projekti za zagotovitev učinkovitejšega trženja izdelkov in storitev (npr. zagotovitev njihovega hitrejšega prihoda na trg).

V nadaljevanju v kratkem podajamo izkušnje in pričakovanja podjetij v zvezi z naložbami v sisteme za upravljanje konfiguracij izdelkov. Nato je v kratkem podanih nekaj ključnih značilnosti oziroma dejavnikov, ki jih je potrebno upoštevati v tovrstnih projektih.

i. Pričakovanja organizacij oziroma zakaj podjetja vlagajo v sisteme za upravljanje konfiguracij izdelkov

Zelo veliko organizacij se v današnjem času srečuje s problemom vse kompleksnejše vsebine svojih izdelkov, pri čemer le s težavo nadzorujejo programsko opremo, ki običajno v največji meri prispeva k zapletenosti tako razvojnega procesa kot same strukture izdelka. Sposobnost hitrega razvoja in spreminjanja izdelka pomeni za podjetje veliko konkurenčno prednost (CMstat, 2002). Z namenom povečanja učinkovitosti podjetja spodbujajo uvajanje hkratnega izvajanja različnih aktivnosti.

Sistemi za upravljanje sprememb in konfiguracij izdelkov v poslovnem okolju so zaščitni znak uspešnih organizacij. Organizacije brez tovrstnih sistemov so običajno manj konkurenčno sposobne v primerjavi z ostalimi. Brez tovrstnih sistemov namreč niso sposobne v ustrezni meri povezati prispevka ljudi, učinkovitih poslovnih procesov, sodobnih programskih okolij in ažurnih informacij (MERANT, 2001, str. 1). Mnoga velika podjetja (Boeing, Xerox, Siemens, General Electric, Texas Instruments itd.) so npr. sredi devetdesetih let vložila v PDM tehnologijo desetine milijonov ameriških dolarjev. Njihove pozitivne izkušnje so vzpodbudile pričakovanja in naložbe tudi pri konkurenci. Organizacije pričakujejo, da bodo z investicijo v tovrstne sisteme dosegle predvsem naslednje učinke (Hewlett-Packard, 2002):

- *Skrajšan cikel prihoda na tržišče (angl. Time-to-Market).* To naj bi bila ena največjih prednosti CM sistemov in je povezana s povečanjem učinkovitosti inženirjev pri razvoju izdelkov. V preteklosti naj bi inženirji velik del (tudi do četrtine) svojega delovnega časa porabili za iskanje in pridobivanje potrebnih informacij o izdelkih. To je vključevalo: iskanje ustrezne informacije s strani uporabnika; operacija pridobivanja gradnika iz odlagališča gradnikov; izdajanje in arhiviranje novih gradnikov itd. Pogosto se je tudi pojavljal t.im. "sindrom ponovnega odkrivanja kolesa" in drugih pojavov nepotrebnega dela v vseh fazah življenjskega cikla izdelka - predvsem iz razloga, ker je iskanje

obstoječe informacije o izdelku lahko zahtevalo več časa kot ponovno reševanje problema. CM sistem pri tem pomaga predvsem s tem, da so: (a) vsi ustrezni podatki o izdelku stalno na voljo vsem udeležencem projekta, ki jih potrebujejo pri svojem delu in imajo ustrezna dovoljenja; (b) z učinkovitim nadzorom sprememb in podpiranjem hkratnega izvajanja nalog.

- *Boljša izraba ustvarjalnosti skupin za načrtovanje izdelkov* je v določeni meri povezana s povečano učinkovitostjo dela. Načrtovalci so v mnogih poslovnih sistemih pogosto "pritisnjeni ob zid" s časovnimi zahtevami, ki jih vsiljuje poslovno okolje - projektni roki, roki za posamezne aktivnosti ipd. Zato večinoma pri načrtovanju uporabljajo znane, konzervativne pristope. Za nove, morda učinkovitejše in kvalitetnejše pristope k reševanju problemov običajno "nimajo časa", saj se bojijo, da bi imel kakršen koli neuspeh prevelike in morda nesprejemljive posledice za uspešnost projekta. Novejši CM sistemi dejansko zmanjšujejo tveganje v zvezi z morebitnimi napakami pri analizi, načrtovanju in razvoju proizvodov. Z uporabo mehanizmov, ki omogočajo skupini transparentni način dela na izdelku, so vsi ostali razvojni inženirji takoj seznanjeni z novimi pristopi in idejami. S tem je preprečeno, da bi morebitna napaka prišla na dan šele v naslednjih aktivnostih ali celo v poznejših fazah projekta, kar bi precej bolj neugodno vplivalo na stroške izdelka. S tem so dejansko presežene komunikacijske in organizacijske ovire in nastanejo večje možnosti za izboljšave.
- *Višja kakovost izdelka.* Pomembna prednost PDM sistemov je, da vsak sodelujoči na projektu lahko dela na isti množici vseskozi ažurnih podatkov. Če inženir npr. spreminja gradnik, natanko ve, kako se to odraža na sistemu. Če npr. pregleduje referenčno kopijo gradnika, se zaveda, da je to kopija ustrezne verzije danega gradnika. Tako je preprečeno, da bi se delo in vsebine gradnikov prekrivali ali da bi prihajalo do neusklajenosti, kljub temu, da zaposleni ves čas delajo hkrati na istem izdelku (lahko celo na isti funkciji ali isti komponenti izdelka). S tem in drugimi prijemi (npr. možnost kakovostnejše izvedbe presoj konfiguracij izdelka) je možno zmanjšati število problemov, ki bi se morebiti pokazali šele v fazi realizacije, preverjanja, proizvodnje ali celo na terenu. To vpliva ne le na višjo kakovost razvoja (po načelu "napravi dobro že v prvem poskusu") in izdelave proizvoda, temveč tudi na že omenjeni čas prihoda na tržišče.
- *Boljši nadzor nad projekti.* Razlog za kasnitev projektov marsikdaj ni v tem, da so slabo planirani, temveč v tem, da je nadzor nad njimi vse težavnejši. Zgodi se npr., da obseg dokumentov in drugih gradnikov v projektu preseže tisto kritično mejo, ki še zagotavlja učinkovito vodenje projektov z obstoječimi pristopi in tehnikami upravljanja. Večji kot je časovni pritisk tekmovalnosti, večja je možnost za pojavitev neusklajenosti, večja je verjetnost, da bo potrebno ponovno izvajati enkrat že (ne najbolje) opravljeno delo. PDM sistemi omogočajo nadzor nad projektom s tem, da zagotavljajo dober nadzor nad podatki, od katerih je odvisen uspeh projekta. Ključni pri tem so npr. struktura in konfiguracije izdelkov, upravljanje sprememb, upravljanje zahtev, sledljivost v projektu ipd. Nadzor lahko poleg tega izboljšamo še z drugimi prijemi, kot npr. z avtomatskim izdajanjem določenih vrst podatkov in elektronskim podpisom.

Poleg koristi, ki jih prinašajo sistemi za upravljanje konfiguracij izdelkov, obstajajo tudi nevarnosti. Ena od teh je (Humphrey, 1989, str. 114), da na te sisteme začnemo gledati le kot na orodja za upravljanje ali kot na pogodbeno obveznost. V tem primeru lahko postanejo birokratska ovira. Ta orodja morajo predvsem tudi podpirati izvajalce pri nadzoru lastnega dela.

ii. Ključni dejavniki v projektih uvajanja CM sistemov

V praksi se je na podlagi izkušenj pokazalo, da imajo pri vpeljavi sistemov za podporo upravljanja konfiguracij izdelkov in njihove programske opreme posebno veliko vlogo naslednji dejavniki:

- kultura poslovnega sistema;
- poslovni procesi;
- tehnologija sistemov za podporo poslovanja in povezovanje teh sistemov v učinkovito celoto;
- finančni dejavniki.

Ker so projekti vpeljave sistemov za upravljanje konfiguracij v veliki meri povezani s spremembo kulture poslovnega sistema in spremembo poslovnih procesov, v strokovni literaturi razmeroma pogosto naletimo na opozorila, da tovrstnih projektov ne moremo uvrščati v kategorijo klasičnih projektov uvajanja informacijskih tehnologij, temveč da gre v tem primeru za kompleksnejšo spremembo.

ii.1 Kultura poslovnega sistema

Uvedba CM sistema v poslovni sistem oziroma način, kako pripraviti uporabnike, da bodo ta sistem sprejeli in učinkovito uporabljali, je predvsem kulturen, sociološki izziv.

Na novo vpeljani sistemi se običajno v večji ali manjši meri razlikujejo od predhodnih. Razmeroma pogosto se pri uvajanju tovrstnih novih sistemov pri uporabnikih pojavijo znaki pasivnega odpora do sprememb (Stevens, 2000, str. 49). Pri tem naj bi bile posebne težave pri uvajanju novih sistemov v poslovnih okoljih, kjer prevladujejo kolektivistični kulturni vzorci, z visoko stopnjo izogibanja negotovosti (Shore, 2000, str. 150). En najpogostejših vzrokov za ovire in nerazumevanje uporabnikov pri uvajanju tovrstnih sistemov je pomanjkanje razumevanja njihovih prednosti (Fries, 1995). Gre torej tudi za komunikacijski problem. Pomembno je, da je vodstvo podjetja v takšnih primerih sposobno nepristransko in vizionarsko oceniti prednosti novega sistema v primerjavi s starim in v sprejemljivi meri prilagoditi način poslovanja tako, da bodo dane možnosti in priložnosti kar najbolje izkoriščene. Potrebno je zagotoviti, da ključni ljudje obiščejo ustrezna usposabljanja, seminarje in preberejo "bele knjige", ki obravnavajo prednosti CM sistemov.

En od učinkovitih pristopov je tudi v tem, da projektno skupino, ki bo uvajala novosti, sestavimo tako, da bo v njej prisoten vsaj po en predstavnik iz vsakega od oddelkov, na katere bodo novosti vplivale. Izkušnje kažejo, da je pred začetkom projekta potrebno pridobiti strinjanje glede vsebine projekta, tako s strani običajnih uporabnikov kot s strani vodstva. Potem, ko pri uporabnikih prevlada zavedanje, da bodo potrebne izboljšave v procesu in orodjih, se z upoštevanjem obstoječih zahtev izoblikujejo cilji in roki projekta, v katerem bodo te spremembe izvedene. Pri izvajanju projekta je pomembno, da se poskusijo doseči zastavljeni cilji in da izvedba ni upočasnjena ali celo ustavljena zaradi pojavljajočih se težav. Večino težav v tovrstnih projektih je namreč mogoče vnaprej predvideti, npr. s pravočasno pridobitvijo ustreznega znanja in informacij, v plan projekta pa vključiti načine njihovega reševanja.

ii.2 Poslovni procesi

En od največjih izzivov v zvezi s podporo poslovnih procesov v podjetju je vsekakor preoblikovati in avtomatizirati te procese ter jih povezati ne le na nivoju podjetja, temveč širše, z dobavitelji, poslovnimi partnerji in ne nazadnje s kupci (Stevens, 2000, str. 48). Pri izbiri CM sistemov so poslovni cilji pomembnejši od tehnoloških. Uporabniki se hitro navdušijo nad najnovejšo tehnologijo, vendar pa se ta stalno spreminja. Tudi zato je potrebno najprej preveriti in po možnosti izboljšati proces, šele nato uvajati ustrezno tehnično podporo za proces.

V fazi preizkušanja prototipa sistema lahko preizkušamo tudi nove metode in postopke na račun starih, vključno s preizkušanjem ustreznih vmesnikov. Uspešnost pri iskanju boljših rešitev običajno zelo motivira skupine uporabnikov, hkrati pa nove rešitve zagotavljajo večjo učinkovitost njihovega dela. Tudi s tem lahko pridobimo uporabnike za sodelovanje in zagotovimo večjo uspešnost projekta uvajanja sistema.
Vedno, preden poskušamo avtomatizirati nek proces ali določen del procesa, je potrebno oceniti, na katerih mestih in kako se le-ta lahko izboljša. V ta namen se je potrebno učiti oziroma tekoče seznanjati s potekom procesa ter shematsko beležiti njegov potek. Na podlagi tega lahko pravočasno izvedemo reinženiring na tistih delih procesa, ki niso optimalni.

ii.3 Tehnologija sistemov za podporo poslovanja in povezovanje teh sistemov v učinkovito celoto

V mnogih okoljih se pomen tehnologije močno precenjuje in se jo postavlja pred poslovni vidik. Po drugi strani pa obstaja nevarnost, da tehnologijo podcenjujemo, kar ima lahko prav tako negativne učinke. Vsekakor je pri sistemih potrebno upoštevati naslednje tehnološke dejavnike:

- objektna zasnova sistema, ki bo na učinkovit način podpirala poslovne objekte;
- ustrezna zgradba sistema, ki dobro podpira delo na različnih geografskih lokacijah;
- možnost podpore projektov, ki sčasoma rastejo;
- možnost vgradnje varnostnih mehanizmov v sistem, itd.

Potencial sistemov za podporo upravljanja konfiguracij izdelkov lahko dosežemo le, v kolikor jih uspemo povezati z drugimi sistemi za podporo poslovanja v kar se da enoten in učinkovit sistem. Infrastruktura, ki omogoča učinkovito povezovanje poslovnih aplikacij, ki lahko tečejo na različnih operacijskih sistemih, predstavlja veliko strateško prednost podjetja. Podjetje ima s tem možnost hitrejšega, enostavnejšega in cenejšega vpeljevanja novih tehnologij. Pri tem ni pomembno le povezovanje sistemov znotraj podjetja, vse pomembnejše je tudi povezovanje s poslovnimi partnerji, dobavitelji, uporabniki. Za integracijo poslovnih aplikacij in sistemov se najpogosteje uporabljata povezovalna programska oprema, ki entitetam aplikacij in podatkovnih zbirk omogoča medsebojno komuniciranje (npr. RPC, povezovalna PO za baze podatkov, aplikativni strežniki...) in standardi za povezovanje poslovnih aplikacij - npr. XML in BizTalk (Šmid, 2001, str. 192). Ti standardi omogočajo enotno predstavitev strukturiranih podatkov in metapodatkov podjetja in njihovo boljše izkoriščanje (Booch, 2000, str. 3).
PDM sistemi morajo poleg učinkovitih povezav s proizvodnimi podpornimi sistemi (npr. ERP sistemi, ki za proizvodnjo uporabljajo kosovnice in sestavnice, ustvarjene v PDM sistemu) in drugimi sistemi za podporo poslovanju posebej v fazah razvoja in vzdrževanja zagotavljati tudi povezavo z razvojnimi okolji in aplikacijami (npr. CASE/CAD/CAM sistemi). To dodatno povečuje učinkovitost procesa razvoja in vzdrževanja izdelkov, s tem pa tudi hitrejše reagiranje na tržne priložnosti.

ii.4 Finančni dejavniki

Finančne ocene projekta se običajno zagotove z ustrezno analizo, kjer se upoštevajo in ovrednotijo koristi in stroški v zvezi z naložbo. V praksi se za projekte uvajanja CM sistemov zelo pogosto uporablja izračun količnika *ROI* (krat. za Return on Investment), ki pove, koliko denarnih enot dobimo v zameno za eno denarno enoto, ki smo jo vložili v nov sistem oziroma v celoten projekt vpeljave sistema, uporabljajo pa se tudi izračuni drugih pokazateljev (Efstathiou, 2002, str. 2). Izračun kazalca ROI temelji na računanju razmerja dobrobiti naložbe nasproti vloženim sredstvom - običajno se sredstva vlože, še predno se pokažejo koristi naložbe. Stroški in koristi se nato običajno prevedejo na sedanjo vrednost denarja. Če poenostavimo: bolj ko čakamo, kdaj se bodo pojavile prednosti naložbe, manj so te vredne, če jih ocenjujemo danes.

Prednosti in stroške, prevedene na sedanjo vrednost, nato seštejemo, da dobimo čisto sedanjo vrednost, in izračunamo njuno razmerje. Stroške, povezane z uvedbo CM sistemov, lahko razdelimo na (MERANT, 2001, str. 4): stroške strojne oziroma materialne opreme, programske opreme, stroške izvedbe, urjenja (trening, šolanje) in stroške nadaljne podpore. Prednosti, ki jih ima uvajanje orodij za upravljanje konfiguracij in sprememb izdelkov v poslovnem okolju, lahko razdelimo na dve skupini: na neposredne in lažje merljive ter na posredne, ki jih težje merimo (MERANT, 2001, str. 2):

- *primeri merljivih prednosti*: glede na začetno stanje okolja, v katerem se razvije sistem za upravljanje sprememb in konfiguracij v poslovnem okolju, le-ta lahko: izboljša produktivnost razvijalca (npr. za 30%); izboljša produktivnost projektnih in funkcionalnih vodij ter izvajalcev testiranja; skrajša razvojne cikle in cikle sprememb; skrajša čas gradnje PO; zmanjša število napak v PO in na spletnih straneh, ipd.;
- *primeri posrednih prednosti*: preprečitev izgube zaradi predolgega razvojnega cikla, nezadovoljenih zahtev kupca, napak v PO izdelka in slabe kakovosti proizvoda; izboljšani načini poročanja v projektih; izboljšana komunikacija in koordinacija med razvojnimi ekipami na različnih geografskih lokacijah; izkoriščanje že izvedenih investicij v razvojna in druga okolja in aplikacije, npr. zaradi učinkovitih aplikativnih vmesnikov do teh orodij; pouporaba obstoječe kode in zmanjšanje ponovljivih razvojnih operacij, itd.

Problematika ocenjevanja in upoštevanja dolgoročnejših in posrednih koristi se pokaže posebej pereča pri najboljših in najdražjih sistemih, npr. ClearCase-u (Crnkovic, 1999, str. 61). V primerjavi s takšnimi sistemi cenejša orodja praviloma na kratki rok prinesejo boljše razmerje med koristjo in vloženimi sredstvi. To je lahko zavajajoče, saj se razmerje na dolgi rok običajno spremeni v korist kakovostnejših in dražjih sistemov.

Ocenjevanje kazalca ROI je razmeroma dober pokazatelj uspešnosti naložbe v danem okolju, saj upošteva kumulativne koristi uporabe sistema za izbrano obdobje, hkrati pa tudi časovno vrednost denarja. Pomembna šibka točka pri izračunih ROI je omejena natančnost izračuna oziroma ocene posrednih in dolgoročnih koristi.