UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

MARTIN VERSTOVŠEK

UPORABA ORODIJ ZA VODENJE PROJEKTOV IT V MAJHNI RAZVOJNI SKUPINI

DIPLOMSKO DELO NA VISOKOŠOLSKEM STROKOVNEM ŠTUDIJU

Mentor:
viš. pred. dr. Damjan Vavpotič

Ljubljana, 2011

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

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

Št. naloge: 00089/2011
Datum: 01.04.2011

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

Kandidat: **MARTIN VERSTOVŠEK**

Naslov: **UPORABA ORODIJ ZA VODENJE PROJEKTOV IT V MAJHNI RAZVOJNI SKUPINI**

USE OF TOOLS FOR MANAGEMENT OD IT PROJECTS IN A SMALL DEVELOPMENT TEAM

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Preučite in predstavite uporabo sodobnih orodij, ki so namenjena učinkovitemu vodenju projektov IT v majhnih skupinah, ki se ukvarjajo z razvojem informacijskih sistemov. V nalogi se posvetite tako teoretičnim kot praktičnim vidikom uporabe orodij ter predstavite njihove prednosti in slabosti. V okviru diplomske naloge predstavite tudi primer takega orodja.

Mentor:

viš. pred. dr. Damjan Vavpotič

Dekan:

prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

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

diplomskega dela

Spodaj podpisani Martin Verstovšek,

z vpisno številko 63030097,

sem avtor diplomskega dela z naslovom:

Uporaba orodij za vodenje projektov IT v majhni razvojni skupini

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom

  viš. pred. dr. Damjana Vavpotiča,

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

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

V Ljubljani, dne 20.07. 2011 Podpis avtorja:

Zahvala

Najprej bi se rad zahvalil mentorju viš. pred. dr. Damjanu Vavpotiču za pomoč pri izdelavi diplomskega dela. Zahvaljujem se tudi vsem, ki so mi skozi celotno obdobje študija na kakršen koli način pomagali. Še posebej bi se zahvalil staršem, ki so mi omogočili študij in mi stali ob strani.

KAZALO VSEBINE

KAZALO SLIK

KAZALO TABEL

SEZNAM UPORABLJENIH KRATIC

IS – Informacijski sistem

IT – Informacijske tehnologije

RTC – Rational Team Concert

IBM – International Business Machines

JTS – Jazz Team Server

JFS – Jazz Foundation Services

HTTP – Hypertext Transfer Protocol

CSV – Comma separated values

POVZETEK

Razvoj programskih rešitev na področju informacijskih tehnologij je navadno projektno organiziran. Za vodenje projekta je zadolžen projektni vodja. Vodenje projektov, kjer se razvijajo nove programske rešitve, je lahko zelo zahtevna naloga.

V diplomski nalogi je v uvodnem delu predstavljeno, kaj smatramo za projekt in kakšne vrste projektov poznamo. V nadaljevanju je povzetih nekaj težav, s katerimi se srečajo vodje projekta pri vodenju. Težave so povzete na podlagi sodelovanja pri razvoju programske rešitve v manjši razvojni skupini.

Osrednji del diplomske naloge predstavlja opis, kako lahko uporaba računalniškega orodja za pomoč pri vodenju projekta pomaga vodji projekta pri vodenju. Podrobneje je predstavljeno računalniško orodje Rational Team Concert. Opisane so njegove funkcionalnosti, ki bistveno pomagajo pri vodenju projekta.

Uporaba računalniškega orodja pomaga vodji projekta. Rezultat pomoči se lahko izrazi kot sprejemanje boljših odločitev vodje projekta, hitrejši prenos informacij med člani projekta, izboljšana celotna organizacija projekta in večja produktivnost celotne ekipe.

Ključne besede: vodenje projekta, računalniško orodje, Rational Team Concert

ABSTRACT

Software solutions development in information technologies is usually led by a project manager. Project manager holds the responsibility for the project and its management. Managing a project can be a very difficult task, especially in the process of developing new software solutions.

In my diploma thesis I first present the meaning of the word project and explain the types of projects known. Then follows a summary of a few main issues that are present in the process of project managing. These issues were recognized in my cooperation in the process of developing a software solutions with a small program team.
The core of my diploma thesis is a description of development tools that would help project managers with their managing. I continue with representation of The tool Rational Team Concert in detail, with description of its functionality, that can significantly help project managers with their work.

The use of computer tool is a great help for the project managers and it's useful assistance to get better results. It's a main tool that helps project managers to consequently make better decisions, fasten the transfer of information between members of the team, and finally, the whole organization of the project stands on a higher level plus the team is more productive.

Key words: project management, computer tool, Rational Team Concert

1. UVOD

Vsebina diplomskega dela se bo gibala okoli vodenja projekta pri razvoju programskih rešitev. Predstavljena bodo računalniška orodja, ki nudijo pomoč pri vodenju projekta informacijskih tehnologij (IT) in kako lahko njihove lastnosti ter funkcionalnosti pomagajo pri vodenju.

Tema je bila izbrana predvsem zaradi pridobljenih izkušenj iz praktičnega izobraževanja. Med samo prakso sem imel priložnost spoznati delovanje manjšega podjetja, ki razvija programske rešitve po naročilu. Sodeloval sem pri projektu, ki je zahteval več ljudi. Tako sem se seznanil z delom v manjši razvojni ekipi in z vodenjem projekta. Sama organizacija te projektne skupine ni bila slaba, vendar mislim, da je še nekaj prostora za izboljšave.

Na začetku je na kratko opisano, kaj pojmujemo pod besedo projekt in pojasnjeno, kakšne načine razvoja rešitev IT poznamo. V nadaljevanju sledi opis opaženih pomanjkljivosti ter slabosti pri vodenju projekta, pri katerem sem sodeloval. To bo služilo kot osnova za jedro diplomske naloge, kjer je opisano, kako lahko izboljšamo proces vodenja projekta. Tu mislim na izboljšanje ob uporabi računalniških orodij, ki pomagajo projektnemu vodji pri vodenju projekta. Orodja omogočajo veliko funkcionalnosti, ki jih brez njih težko nadomestimo, ali pa jih sploh ne moremo.

Obstaja veliko takšnih računalniških orodij. Na trgu lahko dobimo plačljiva in odprtokodna orodja. Podrobneje sta predstavljeni orodji IBM Rational Team Concert in Jira. Opisane so njune funkcionalnosti in prikazano je, kako lahko njihova uporaba pomaga pri vodenju projekta manjše razvojne skupine.

2. PROJEKT IN OSNOVNI PRISTOPI RAZVOJA

2.1 OPREDELITEV PROJEKTA

Projekt je enkratna, zahtevna in kompleksna skupina nalog, ki mora biti končana v določenem roku. Pri tem mora projekt doseči vnaprej zastavljene cilje in omejitve.

Ko govorimo o tem, da je projekt enkraten, s tem mislimo, da ne gre za ponavljajoči proces, ampak za enkratno skupino nalogo, ki se v enaki obliki opravi samo enkrat. Taka skupina nalog se ne izvaja trajno, niti ne ponavlja v znanih časovnih intervalih.

Projekti so večinoma zahtevna in kompleksna skupina nalog. Zahtevnost se lahko izraža preko velikega števila ljudi, ki sodeluje pri projektu. Veliko ljudi je potrebno, kadar gre za večjo obsežnost delovnih nalog in uporabo več različnih tehničnih pripomočkov. Pri tem je potrebna dobra koordinacija med vsemi sodelujočimi. Prav tako se zahtevnost izrazi v primeru, kadar naloge projekta od nas zahtevajo posebej poglobljena specifična znanja.

Če je projekt sestavljen iz več delov, je potrebno najprej ugotoviti medsebojno povezanost med njimi, čeprav se zdijo na videz samostojni. Ko odkrijemo medsebojno odvisnost in določimo temeljne dele projekta, s pomočjo katerih definiramo vrstni red razvoja, smo uspešno obvladali kompleksnost projekta.

Na rok dokončanja projekta vpliva več dejavnikov. Odvisen je lahko od aktualnosti naloge, razpoložljivih zmogljivosti za izvedbo in tudi od motivacije ljudi, ki delajo na projektu. Rok dokončanja projekta lahko narekujejo tudi zakonski predpisi, tržne razmere, konkurenca in strategija podjetja.

Cilji morajo biti določeni že pred začetkom projekta. Služijo nam za kontrolo napredka med razvojem, kot motivacija pri težavah in so merilo novih pridobitev, ki jih dosežemo ob dokončanju projekta. Zastavljeni cilji nam ob koncu projekta služijo kot kontrolne točke, katere od zastavljenih ciljev smo dejansko dosegli.

Najbolj pogosta omejitev pri projektu so stroški, ki nastanejo tekom razvoja projekta, kadri za delo na projektu ter strojna in programska oprema. Omejitve pri projektu nam lahko predstavljajo pravila, predpisi, standardi, navade, poslovna filozofija, lahko pa tudi splošna kultura okolice, v kateri se nahajamo.

2.2 VRSTE PROJEKTOV

Projekte lahko delimo na podlagi različnih kriterijev. Kot kriterij lahko vzamemo cilj, namen, način izvedbe, trajanje, kompleksnost ali lokacijo projekta.
Kot temeljni kriterij za delitev vzemimo cilj projekta. Če se osredotočimo na cilj, razdelimo projekte na dva dela. Deterministične in stohastične. Poznamo še eno osnovno delitev projektov. Ta jih razdeli na enkratne projekte in projektne procese.

2.2.1 DETERMINISTIČNI IN STOHASTIČNI PROJEKTI

Deterministični projekti so tisti, pri katerih lahko končne cilje povsem določimo. Kadar imamo natančno določene končne cilje, lahko na podlagi teh določimo delne cilje in ciljno retrogradno oblikujemo projekt. To pomeni, da nam jasno določen končni cilj omogoča, da na njegovi osnovi določimo vse aktivnosti, ki so potrebne za dosego tega cilja in s tem dokončanje projekta. Za deterministične projekte je tudi značilno, da so njihovi končni cilji uresničljivi z veliko verjetnostjo.

Med skupino stohastičnih projektov spadajo vsi tisti, pri katerih končnih ciljev ni mogoče natančno definirati. Sem največkrat uvrščamo razvojne in raziskovalne projekte. Zanje je značilno, da nam njihovi delni rezultati začetnih aktivnosti narekujejo in pomagajo definirati nadaljnje cilje. Oblikovanje projektov na tak način imenujemo ciljno progresivni. Primer stohastičnega projekta je raziskava v laboratoriju.

2.2.2 ENKRATNI PROJEKTI IN PROJEKTNI PROCESI

Enkratni projekti so tisti projekti, ki se pojavijo samo enkrat. Projekt v takšni obliki se ne bo več nikoli ponovil, niti se ni pojavil že v preteklosti. Takšni projekti zahtevajo namensko zasnovano projektno organizacijo, ki se razlikuje za vsak enkratni projekt. Primer projekta, ki spada med enkratne projekte, je izgradnja viadukta.

Projektni procesi se v podobnih okoliščinah večkrat ponovijo. To so projekti, ki imajo enake ekonomske ali tehnološke značilnosti. V gradbeništvu najdemo značilne primere takih projektov. Zahtevajo ustaljen način vodenja in izvedbe, zato je njihova projektna organizacija stalna in se ne spreminja. Takšen projekt je lahko gradnja hiše.

2.3 NAČINI PRISTOPOV K RAZVOJU PROJEKTA

Način, kako bomo razvijali in vodili razvoj posameznega projekta, je odvisen od več dejavnikov. V prvi vrsti od vrste projekta. Izbrali si bomo model razvoja, ki je najbolj primeren za naš projekt. Tako bomo lahko izkoristili vse prednosti izbranega modela in učinkovito pripeljali projekt do končnega cilja. Vpliv na izbor modela, po katerem bomo razvijali, ima tudi sam način dela razvojne ekipe. Tu mislim na izkušnje razvojne ekipe s posameznim razvojnim modelom, ki ga je uporabljala že v preteklosti.

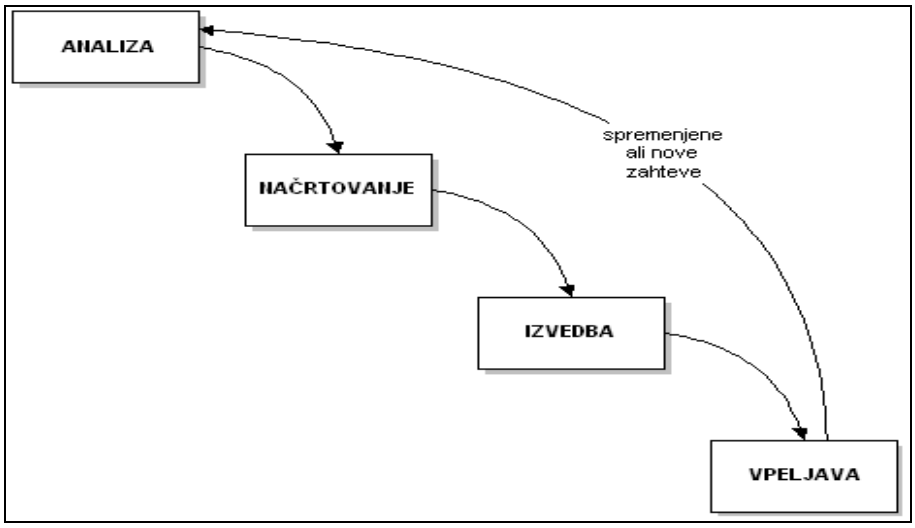
Ker je razvoj projekta običajno skupek zahtevnih nalog, si pomagamo tako, da razvoj in vodenje projekta razdelimo na posamezne faze ali dele projekta. Poznamo naslednje faze projekta:

- analiza,
- načrtovanje,
- implementacija,
- testiranje,
- uvedba in
- vzdrževanje.

V teoriji poznamo več načinov, modelov razvoja, ki ga poimenujemo tudi življenjski model razvoja informacijskega sistema (IS). Razvoj in vodenje projekta sledi življenjskemu ciklu oziroma razvojnemu modelu, ki določa zaporedje faz razvoja. Torej, življenjski model nam pove, v kakšnem zaporedju in na kakšen način si sledijo posamezne faze. V praksi se večinoma uporablja kombinacija različnih modelov.

2.3.1 ZAPOREDNI ALI SLAPOVNI MODEL

Zaporedni ali slapovni model je najstarejši razvojni model. Vse aktivnosti si sledijo zaporedno. Naslednja faza se začne šele tedaj, ko se prejšnja v celoti zaključi. V zaporednem modelu se ni možno vrniti na predhodne faze, zato je nujno, da so zahteve že na začetku natančno in nedvoumno definirane. Pri tem je zelo pomembno, da na začetku pri analizi zelo tesno sodelujeta izvajalec projekta in naročnik. Edino na tak način lahko preprečimo, da pri razvoju ne pride do prevelikega odstopanja od ciljev projekta.
Zaporedni model je primeren za relativno kompleksne projekte, če se zahteve med samim razvojem ne spreminjajo, in za rutinske projekte, še posebej tedaj, ko razvija projekt izkušena skupina razvijalcev. Omogoča tudi natančno projektno vodenje, saj lahko načrtovanje v celoti izvedemo vnaprej.
Ena od slabosti zaporednega modela je nefleksibilnost. Ta pride do izraza, kadar se spremeni kakšna zahteva, saj je potrebno veliko truda, da se popravi ali dopolni kakšna malenkost. Slapovni model ne omogoča vzporednega razvoja. To potrjuje tudi praksa, saj težko pričakujemo, da se celoten postopek v celoti zaključi še preden se začne naslednji.

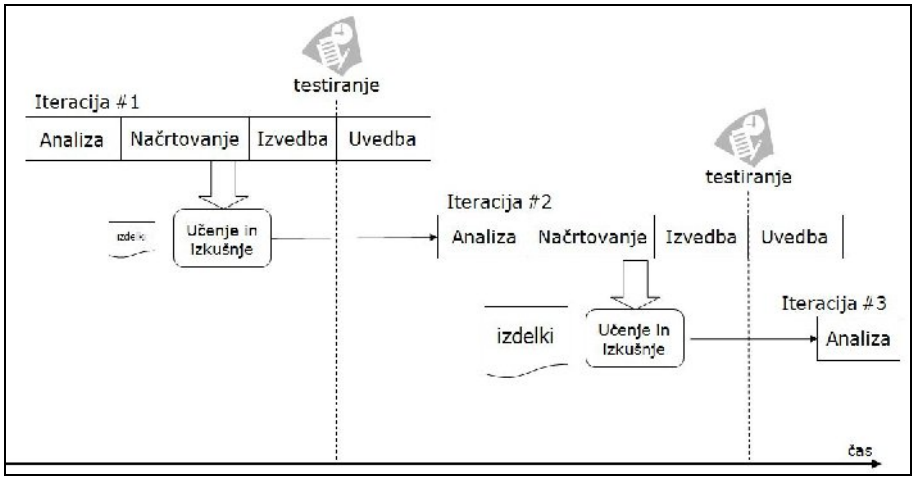


Slika 1: Zaporedni ali slapovni model

2.3.2 ITERATIVNI MODEL

Pri iterativnem modelu faze razvoja izvajamo v več iteracijah. Iteracija je specifično zaporedje aktivnosti, izvedenih na osnovi načrta in z določenim kriterijem vrednotenja, ki se konča z izdajo izdelka. Model je bil razvit kot odgovor na pomanjkljivosti zaporednega modela. Posebnost iterativnega modela je v tem, da v vsaki iteraciji razvijemo samo en del funkcionalnosti celotnega sistema in ne razvijamo cel sistem naenkrat. Posamezna iteracija gre skozi vse faze razvoja, vendar ne z enako intenzivnostjo. Naslednja iteracija se lahko začne šele takrat, ko se prejšnja v celoti zaključi. Med izvajanjem ene iteracije ne dopustimo, da bi se uvedla kakšna sprememba. Vse spremembe se vpeljejo na začetku posamezne iteracije.
V začetnih iteracijah razvijamo najbolj rizične dele sistema in s tem zmanjšamo tveganje celotnega projekta. Lepa lastnost iterativnega modela je tudi zgodnja povratna informacija

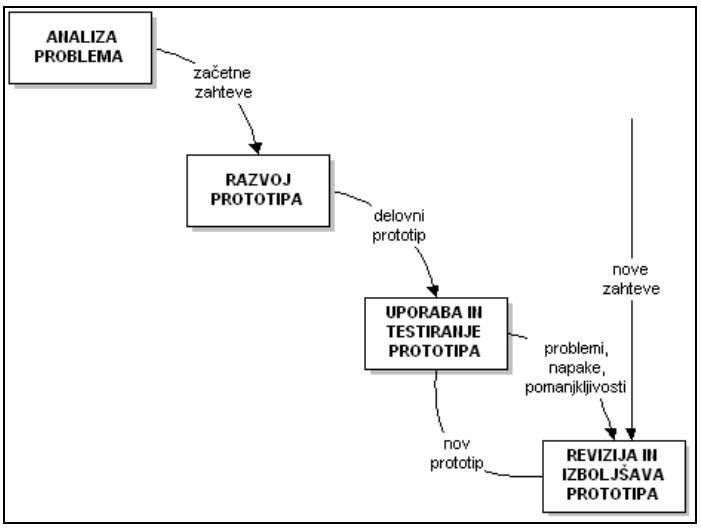
uporabnikov, saj že po prvi iteraciji dobimo izdelek. Pri takem modelu razvoja lahko tudi predamo del projekta, še preden je v celoti končan.
Iterativni model ne omogoča dobrega načrtovanja poteka projekta, saj ni mogoče natančno definirati, koliko iteracij bo potrebno, da bomo razvili dovolj dober končni izdelek. Zato je vodenje projekta in vzdrževanje dokumentacije z iterativnim pristopom zelo zahtevno.



Slika 2: Iterativni model

2.3.3 PROTOTIPNI MODEL

Pri prototipnem modelu gre v bistvu za različico iterativnega pristopa. Model temelji na izdelavi prototipov. Prototip opredelimo kot predhodno izdelan, navadno še nepopoln primerek sistema ali dela sistema. V postopku razvoja ločimo dve osnovni fazi. To sta izdelava in ocenjevanje prototipa, ki se lahko večkrat ponovita. Vsak prototip pa poskušamo izboljševati, dokler ne izdelamo končne rešitve, ki zadovolji zahteve po kakovosti in izpolni načrtovane cilje.
Prototipni pristop uporabimo takrat, kadar na začetku projekta težko popolno definiramo zahteve, ali kadar razvijamo povsem novo aplikacijo.



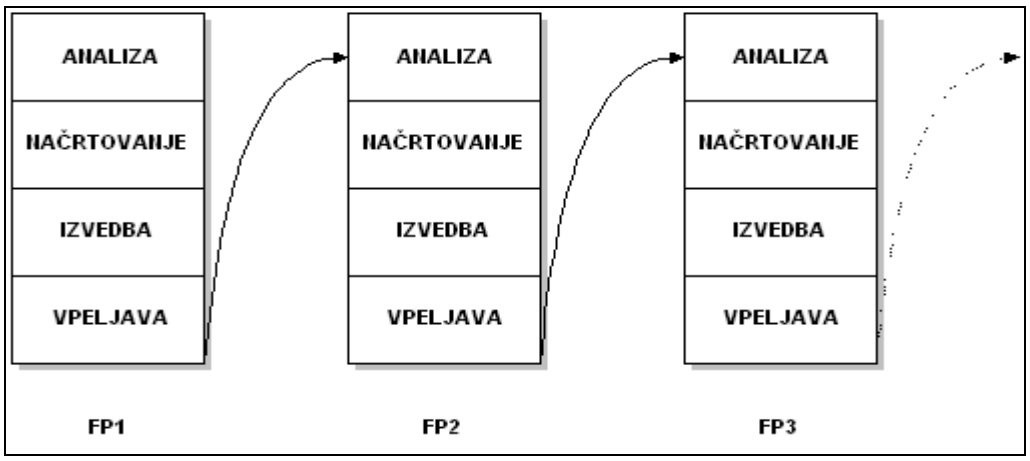
Slika 3: Prototipni model

2.3.4. INKREMENTALNI MODEL

Pri modelu inkrementalnega razvoja razvijamo celotno rešitev po delih. Že razvite dele celotne rešitve sproti predajamo naročniku. Posamezen del rešitve ali inkrement predstavlja zaokroženo funkcionalnost sistema (modul, podsistem, sklop). Tu ne razvijamo celotnega projekta naenkrat. Najprej razvijemo en inkrement v celoti, ga predamo uporabniku in nadaljujemo z razvojem naslednjega sklopa. Seveda je potrebno predhodno določiti prioriteto sklopov, ki se bodo razvijali. Običajno razvijemo najprej sklope, ki predstavljajo osrednjo funkcionalnost celotne rešitve, in sklope, ki predstavljajo največje tveganje. Ob predaji novega sklopa uporabniku ga povežemo z ostalimi sklopi, ki smo jih že predhodno razvili. Tako sestavimo končno rešitev za uporabnika iz več integriranih sklopov.

Prednost inkrementalnega razvoja je v tem, da naročnik hitro dobi del zahtevane celotne rešitve in lahko na podlagi tega, tudi sam sodeluje pri testiranju. Naročnik ima tudi dober pregled nad napredovanjem razvoja celotnega projekta.

Slabost modela se izkaže pri projektih, ki se ne dajo razdeliti na sklope zaradi medsebojne odvisnosti. Pojavijo se tudi težave, če smo pri načrtovanju naredili napako in napačno razdelili celotni projekt na posamezne sklope ter odvisnosti med že razdeljenimi sklopi opazili šele pri kasnejšem razvoju same programske kode.



Slika 4: Inkrementalni model

3. POMANJKLJIVOSTI PRI VODENJU PROJEKTA IZ PRAKSE

3.1 PREGLED KLASIČNIH NAPAK

Napake, ki se lahko pojavijo pri vodenju projekta, lahko razdelimo v štiri glavne skupine. To so napake povezane z ljudmi, s procesi, s produktom in s tehnologijo.

Napake povezane z ljudmi	**Napake povezane s procesom**	**Napake povezane s produktom**	**Napake povezane s tehnologijo**
Oslabljena motivacija	Preveč optimističen razpored	Prepodrobne zahteve	Prevelika pričakovanja ob prvi uporabi novih tehnologij
Šibko osebje	Nezadostno načrtovanje rizikov	Neustrezno obravnavanje sprememb zahtev	Precenjeni prihranki ob uporabi novih orodij in tehnologij
Problematični posamezniki	Pogodbene napake	Razvoj funkcionalnosti, ki niso zahtevane	Zamenjava orodij na sredini projekta
Heroji projekta	Nezadostno načrtovanje poteka projekta	Neustrezno spreminjanje razporeda	Pomanjkanje samodejnega nadzora izvorne kode
Dodajanje ljudi na zamujajoči projekt	Opustitev načrtovanja pod pritiskom	Raziskovalno orientiran razvoj	
Hrupne pisarne	Tratenje časa pred odobrenim začetkom projekta		
Trenja med razvijalci in stranko	Krajšanje potrebnih sprememb		
Nerealistična pričakovanja	Neustrezno oblikovanje		
Premalo učinkovito sponzoriranje projekta	Zmanjševanje zagotavljanja kakovosti		
Premajhen vložek vseh interesnih skupin	Nezadosten nadzor vodstva		
Premajhno sodelovanje končnega uporabnika	Izpuščanje potrebnih nalog pri oceni		
Postavitev politike vodenja pred rezultate	Načrtovati ujeti zaostanek kasneje		
Preveč optimističen pogled	Hitro in ohlapno programiranje		

Tabela 1: Klasične napake pri projektu

Tabela 1 prikazuje kar nekaj klasičnih napak, ki se lahko pojavijo pri vodenju projekta. Nekaterim se lahko vodja projekta izogne zgolj s skrbno premišljenim ravnanjem. Na tak način bi lahko odpravil napake, kot so: oslabljena motivacija, preveč optimističen pogled, slabo delovno okolje itd.

Nezadostno in slabo načrtovanje ter razporejanje delovnih nalog, slaba komunikacija, slab pregled nad stanjem projekta pa so primeri napak, ki bi jih lahko razvojne skupine odpravile, če bi uporabljale ustrezno računalniško orodje. Te vrste napak sem tudi sam zaznal med opravljanjem obvezne delovne prakse v majhni razvojni skupini in tudi kasneje. Zato bom tem napakam v nadaljevanju posvetil več pozornosti. Kot sem že omenil, lahko te napake odpravi uporaba računalniškega orodja. Eno takih je Rational Team Concert (RTC). To pa je drugi razlog, saj bo RTC, ki na eleganten način s svojo funkcionalnostjo odpravi te napake, podrobneje predstavljen v nadaljevanju.

3.2 SLABA KOMUNIKACIJA

Mislim, da je komunikacija in informiranost med sodelujočimi pri projektu ključnega pomena za uspešno delo celotnega tima. Tu bi lahko izpostavil dve različni poti komunikacije. Ena je med naročnikom in vodjo projekta, druga pa med vodjo projekta in njegovimi podrejenimi člani projektne ekipe.

Vsi si želimo, da bi imeli z naročnikom dober odnos in bi z njim lahko sproščeno komunicirali, saj nam le-ta komunikacija predstavlja vir najbolj točnih informacij. Naročnik najbolje ve, kakšno programsko opremo potrebuje. Kot sem opazil pri opravljanju prakse, je komunikacija z naročnikom potekala predvsem preko elektronske pošte, z občasnimi srečanji. Za veliko stvari sta se dogovorila naročnik in vodja projekta preko elektronske pošte. Včasih je prišlo do situacije, da vodja projekta ni posredoval dogovora razvojni ekipi, ali pa je pozabil spremeniti vsebino delovne naloge, ki je zajemala temo novega dogovora. Zaradi tega je občasno nastala zmešnjava, ki je negativno vplivala na celoten projekt.

Zelo pomembna je tudi komunikacija znotraj članov razvojne ekipe in razvojne ekipe s projektnim vodjem. Nujno je, da so člani med sabo pravočasno informirani, saj lahko le na tak način uspešno poteka delo.
Včasih sem kot razvijalec pogrešal direktno komunikacijo z naročnikom. To pa zato, ker bi lahko kakšno vsebinsko vprašanje, ki se pojavi pri implementaciji uporabniških zahtev, zelo hitro razčistili z direktno komunikacijo.

3.3 SLABO NAČRTOVANJE IN DODELJEVANJE DELOVNIH NALOG

Vodja ima zahtevno nalogo pri terminskem načrtovanju projekta in pri dodeljevanju delovnih nalog posameznim članom razvojne ekipe. Skrbeti mora, da so ustrezne funkcionalnosti na voljo v predvidenih časovnih rokih. Za realizacijo tega pa mora ustrezno, glede na prioriteto, razdeliti delovne naloge razvijalcem v ekipi. Težko si predstavljamo, da bi to počel na pamet brez kakšnega pripomočka, saj mora pri tem paziti, da porazdeli delo med ekipo enakomerno in da posameznikov ne preobremeni. Predvidevati mora tudi, kdaj bodo posamezne delovne naloge končane, da lahko oceni, ali bo celotna funkcionalnost dokončana do predvidenega roka. Poleg tega mora upoštevati tudi možne rizike, ki se lahko pojavijo.

Iz svojih opažanj bi izpostavil, da bi bilo razporejanje in dodeljevanje delovnih nalog lahko boljše izvedeno. Prihajalo je do preobremenitev na posameznega razvijalca. Če bi projektni vodja uporabljal orodje, ki bi mu bolj nazorno prikazalo trenutno stanje na projektu, bi lahko bolje razporedil delo.

3.4 SLAB PREGLED NAD STANJEM PROJEKTA

Pregled nad realnim stanjem projekta je ključnega pomena za vodjo projekta. Le na podlagi pravega trenutnega stanja projekta, lahko sprejema pravilne odločitve. Da lahko projektni vodja pridobi realno oceno o trenutnem stanju projekta, mora imeti pregled nad vsemi člani projekta in njihovim že dokončanim delom. To lahko stori tako, da vsakega posebej povpraša, koliko predvidenega dela je že opravil in koliko časa še potrebuje za preostanek. Če ni veliko članov projekta, je lahko tudi taka pot rešitev. Vendar to terja dodatni čas projektnega vodje in tudi čas članov, da mu poročajo o opravljenem delu. Ko zbere informacije, mora pripraviti še poročilo. Če bi lahko to projektni vodja opravil sam, s pomočjo računalniškega programa, bi vsi privarčevali dragoceni čas. Zmanjšal bi se tudi človeški faktor in s tem možnost napačne ocene stanja.

3.5 PREDLOGI ZA ODPRAVO POMANJKLJIVOSTI

Da bi lahko odpravili zgoraj opisane slabosti, bi lahko tudi manjše razvojne skupine v svoj proces dela vključile uporabo računalniškega programa, ki pomaga pri vodenju in razvoju projektov IT. Z uporabo teh orodij bi imel projektni vodja na voljo veliko funkcionalnosti orodij, ki bi mu zelo pomagale pri vodenju. Sam proces vodenja bi se lahko izboljšal, s tem pa tudi produktivnost celotne razvojne skupine.

4. UPORABA RAČUNALNIŠKEGA ORODJA PRI VODENJU IT PROJEKTA

4.1 UVOD

Vodenje projekta in s tem razvoj novega IS je postopek, za katerega lahko najdemo ogromno opisanih, definiranih teoretičnih navodil, ki nam narekujejo, kako izpeljati projekt od začetka do konca. Za področje samega razvoja IS obstaja kar nekaj metodologij razvoja, ki nam natančno narekujejo, kako mora le-ta potekati. Ker so možni načini razvoja opisani že na začetku, se bom v tem poglavju osredotočil na uporabo računalniškega orodja za pomoč pri vodenju IT projekta.

Predstavljene bodo funkcionalnosti, ki jih nudijo ta orodja. Ravno te funkcionalnosti pa vodji projekta omogočajo lažje in bolj učinkovito vodenje. Podrobneje bodo opisani tudi dve taki orodji. Prvo je orodje IBM Rational Team Concert, produkt podjetja International Business Machines (IBM). To je orodje, ki zajema veliko funkcionalnosti, ki pomagajo vodji projekta pri vodenju. Orodje RTC je eden izmed vodilnih predstavnikov orodij iz skupine računalniških programov, ki se uporabljajo kot pripomoček pri vodenju in razvoju IT projektov. Drugo orodje je Jira, ki je produkt avstralskega podjetja Atlassian. Osnovna različica nima tako bogat nabor funkcionalnosti kot RTC, vendar vsebuje vse funkcionalnosti, ki so potrebne za odpravo pomanjkljivosti iz poglavja 3. Na voljo pa je tudi različica Jire z dodanimi vstavki, ki ima tudi zelo bogat nabor funkcionalnosti, ki podpirajo vodenje in razvoj IT projekta.

4.2 SPLOŠNE FUNKCIONALNOSTI ORODIJ

4.2.1 SISTEM ZA SKUPINSKO UPORABO

Računalniška orodja, ki se uporabljajo kot pripomoček pri vodenju IT projekta, so zasnovana kot odjemalec strežnik aplikacija. Na strežniku tečejo storitve orodja, do katerih nato dostopajo odjemalci oziroma uporabniki. Uporabniki lahko dostopajo do strežniškega dela orodja preko več načinov. Lahko dostopajo preko spletnega protokola HTTP ali pa preko uporabniškega vmesnika, ki ga ima posamezno orodje. Način dostopa je odvisen od posameznega orodja, nekatera pa podpirajo obe možnosti. Na strežniku se tudi hranijo podatki.

Takšna zasnova omogoča veliko prednosti, ki jih s pridom izkoriščajo uporabniki. Pri razvoju IT projektov sodeluje več ljudi. Vsak ima svojo vlogo, lahko se združujejo tudi v skupine. Tako lahko obstaja več skupin znotraj projekta. Ravno takšno ureditev zelo dobro podpirajo računalniška orodja za pomoč pri vodenju in razvoju IT projektov, kot so RTC in Jira. Vsak uporabnik ima določeno vlogo. To omogoča projektnemu vodji, da lahko posameznega uporabnika, ali pa uporabnike z enako vlogo podrobno spremlja skozi cel ciklus projekta. Takšna možnost pogleda je ključnega pomena za vodjo projekta in mu omogoči dober pregled nad stanjem celotnega projekta.
Ker sodelujoči na projektu uporabljajo računalniško orodje, kot pripomoček pri razvoju IT rešitve, imajo v vsakem trenutku dostop do najnovejših informacij. S tem mislim dostop do aktualne dokumentacije, zadnje verzije razvojne kode, stanja delovnih nalog in ostalih podatkov, ki so pomembni za posamezen projekt. Na ta način je ekipa, ki dela na projektu, povezana med sabo in znotraj nje se informacije hitro širijo. Bolj kot so udeleženci projekta povezani in informirani, večja je produktivnost in dosegajo se boljši rezultati dela.

4.2.2 VODENJE IN SLEDENJE DELOVNIM NALOGAM

Delovne naloge so osnovna enota, na podlagi katerih vodje projekta ugotavljajo napredek projekta. Omogočajo sledenje delovnemu procesu. Delovne naloge so lahko različnih tipov. Najbolj pogosta sta naslednja dva tipa: naloga in napaka. Vsaka delovna naloga ima tudi svojo življenjsko dobo. Med tem obdobjem pa prehaja med različnimi stanji. Nekatera računalniška orodja tudi omogočajo definicijo stanj za delovno nalogo.

Orodja omogočajo zelo natančno in pregledno sledenje posameznim delovnim nalogam. V ta namen imajo posebej vgrajen iskalnik, s katerim lahko poiščemo posamezno nalogo. Poizvedbo lahko naredimo po enem ali več različnih kriterijih hkrati. Primer kriterija za iskanje delovnih nalog bi lahko bil:

- ime vloge, kateri je bila dodeljena,
- ime delovne naloge,
- stanje delovne naloge,
- tip delovne naloge in
- ime projekta, kateremu spada delovna naloga,
- itd.

Funkcionalnost iskanja delovnih nalog je zelo pomembna. Lahko jo uporabljajo vsi člani projekta. Posebej pomembna je za projektnega vodjo. Ta določa delovne naloge in jih dodeljuje posameznim vlogam na projektu. Razvršča jih tudi v ustrezne skupine. Na podlagi iskanja lahko tudi hitro ugotovi v kakšnem statusu je posamezna delovna naloga, kdo ima dodeljeno iskano nalogo, ali že obstaja določena naloga in po potrebi ukrepa.

Večina orodij omogoča, da lahko na posamezno delovno nalogo zabeležimo tudi dodatne informacije. To so lahko: beleženje lastnih komentarjev ali zapiskov lastnika naloge, dokumentacije, porabljenega časa za izvedbo naloge in še kakšna dodatna možnost, glede na vrsto računalniškega orodja.

4.2.3 UPRAVLJANJE Z IZVORNO KODO

Izvorna koda projekta se hrani na strežniški strani, ločeno od lokalnih različic kode posameznih razvijalcev. Vsak razvijalec svoje spremembe dodaja h kodi, ki se nahaja na strežniku. Orodja pri tem poskrbijo, da se koda, ki se dodaja na strežnik, s strani posameznega razvijalca pravilno doda. Preverja se, ali so v dodani kodi kakšne konfliktne napake. Če se zaznajo konflikti, to pomeni, da sta dva razvijalca hotela dodati različno kodo na enako mesto. V takšnem primeru poskusi orodje samo združiti spremembe. V kolikor mu ne uspe, ustrezno opozori razvijalca, ki dodaja novo kodo. Ta mora nato sam izbrati, kateri del kode se bo prenesel na strežnik. S takim načinom nam orodje pomaga, da se na strežniku nahaja samo pravilna koda. Nekatera orodja omogočajo tudi eksplicitno zaklepanje delov kode. To pomeni, da ta del kode, lahko ureja samo en uporabnik.

Še vedno pa obstaja možnost napake v izvorni kodi. Zato periodično tudi gradimo izvorno kodo, ki gre nato skozi postopek testiranja. Tu se ugotovijo vse morebitne pomanjkljivosti, ki se nahajajo v zgrajeni izvorni kodi. Posamezna orodja imajo zelo dobro podprto grajenje izvorne kode, ki omogočajo načrtovanje grajenja, poročanje morebitnih napak in beleženje statistike grajenja.

Takšen način avtomatičnega zbiranja informacij orodja o grajenju izvorne kode daje vodji projekta pomembne informacije. Vodja se lahko na podlagi teh podatkov odloča o morebitni novi izdaji izvorne kode v primeru, da je brez napak. V nasprotnem primeru pa lahko načrtuje čas, ki je potreben, da se odpravijo vse pomanjkljivosti.

4.2.4 NAČRTOVANJE IN VODENJE PROJEKTA

Kot sem že prej omenil, je delovna naloga osnova enota, na podlagi katere se beleži sledljivost na projektu. Vendar ima še drugo zelo pomembno funkcijo. Delovni nalogi lahko tudi določimo, kdaj se bo pričela in koliko časa je potrebno, da se jo dokonča. Poleg teh dveh podatkov moramo določiti še vrstni red izvajanja delovnih nalog. Ko imamo to definirano za vse delovne naloge, nam program avtomatsko izračuna predvideni čas trajanja projekta. Določali smo tudi vrstni red izvajanja nalog. Na podlagi tega nam večina računalniških orodij izriše Ganttov diagram. Ta nam zelo lepo vizualno prikaže, katere naloge so na kritični poti, ki določa, koliko časa bo najmanj potrebno za dokončanje projekta. Iz tega diagrama se tudi vidi, katere delovne naloge lahko potekajo vzporedno. To je zelo pomemben podatek za vodjo

projekta, saj lahko, ob predpostavki, da ima na voljo dovolj resursov, načrtuje vzporedno izvedbo več delovnih nalog.

Računalniški programi nam ponujajo že kreirane predloge za razvrščanje delovnih nalog, ki lahko temeljijo na tradicionalnem načinu vodenja ali pa zajemajo tehnike agilnega vodenja. Tudi ta funkcionalnost orodij je dobrodošla in v prid vodenju, saj lahko projektni vodja glede na strukturo projekta, svoje lastne želje in na podlagi usmerjenosti organizacije, izbere način, ki je najbolj primeren za posamezen projekt.

4.2.5 POROČILA IN DOKUMENTACIJA

Projektni vodja želi imeti dober pregled nad stanjem projekta in želi vedeti, kam se nagibajo trendi posameznih delov projekta. To informacijo mu računalniško orodje, ki ga uporablja kot pripomoček pri vodenju projekta, zagotovi v zelo kratkem času in v zelo pregledni obliki. Orodja ponujajo možnosti različnih izpisov, ki jih lahko uporabnik pridobi z nekaj kliki preko uporabniškega vmesnika. Na ta način lahko projektni vodja v vsakem trenutku dostopa do aktualnih informacij projekta, z relativno malo vloženega trudi. Ti podatki mu lahko služijo kot podlaga za sprejemanje odločitev. Ker so informacije, ki jih pridobi z izpisi ažurne, so tudi odločitve na podlagi njih dobre. S tem se tudi zmanjša riziko sprejemanja napačnih odločitev in posledično pripomore k dobremu vodenju projekta.

Orodja ponujajo izpis poročil trenutnega stanja. Klasični primer izpisa, ki odraža trenutno stanje, je izpis delovnih nalog. Te lahko posebej razčlenimo na njihove statuse, kot so odprta naloga, napaka ali končana naloga in na ta način oblikujemo poročilo po lastni želji. Druga vrsta so izpisi, ki prikazujejo trend gibanja izbranega parametra. Tako lahko izpišemo poročilo, ki nam pove, kako se giblje količina opravljenega dela na projektu, v primerjavi z načrtovanim delom. V odvisnosti od uporabljenega računalniškega orodja imamo na voljo različne izpise, ki jih to orodje ponuja.

Zelo uporabna je tudi funkcionalnost shranjevanja dokumentacije. Orodja omogočajo, da lahko shranimo kar celotno datoteko, ki predstavlja dokumentacijo ali pa datoteko s kakšno dodatno razlago. Dokumente lahko pripnemo posamezni delovni nalogi. To nam omogoča, da imamo za posamezno delovno nalogo vedno pri roki tudi njeno podrobno vsebinsko razlago. S tem prihranimo veliko časa, saj ne rabimo vedno znova iskati dokumentacijo, ampak jo imamo na dosegu enega klika. To pride do izraza še posebej takrat, ko delovna naloga ni v prvi iteraciji pravilno implementirana.

4.3 ORODJA IN AGILNI PRISTOP

4.3.1 KAJ POMENI AGILNI PRISTOP

Agilni pristop vodenja in razvoja programske opreme se je pojavil kot odziv na zapletene tradicionalne metodologije razvoja. Tradicionalne metodologije razvoja relativno pozno dajejo rezultate in se držijo striktnih pravil posamezne metodologije. Agilni pristop pa je ravno alternativa temu.

Agilni pristop zagovarja naslednje načela, ki so bile postavljene leta 2001 na konferenci v Snowbird-u (Utah, ZDA):

Posameznik in njihova komunikacija so pomembnejši kot sam proces in orodja.

- Potrebno je posvetiti več pozornosti članom projekta, njihovim znanjem in željam.
- Dobra komunikacija je ključ za uspešnost projekta.
- Bolje je, da je komunikacija dobra, čeprav se člani projekta ne držijo predpisanih pravil procesa, kot pa, da se člani držijo pravil in je komunikacija med njimi slaba.

Delujoča programska oprema je pomembnejša kot popolna dokumentacija.

- Delujoč program je največ kar lahko dobi uporabnik.
- Dokumentacija je koristna, vendar vseeno sekundarnega pomena. Pomembno je, da jo izdelujemo s poudarkom »ravno dovolj«.

Vključevanje (sodelovanje) uporabnika je pomembnejše kot pogajanje na osnovi pogodb.

- Odnos med naročnikom in izvajalcem ne sme biti preveč formalen in strog. Pomembno je da se dobro ujameta in ne gojita strogo pogodbenega odnosa, ki lahko zamaje projekt.
- Naročnik najbolje ve, kaj potrebuje, čeprav včasih to ne zna razložiti, zato ga postavljamo na pomembno mesto.

Upoštevanje sprememb je pomembnejše od sledenja planu.

- Naivno je pričakovati, da bomo v začetnih fazah projekta zajeli vse zahteve.
- Projektni plani so koristni, vendar morajo omogočati spremembe, vsaj v smislu prioritet znotraj dogovorjenega okvirja.
- Sledenje planu je koristno, vendar le do stanja, ko se to ne razlikuje preveč od dejanskega. Slabo je slediti planu, ki je že zastarel.

Iz osnovnih načel agilne tehnike so izpeljana tudi naslednja priporočila:

- Najvišja prioriteta je zadovoljiti kupca skozi zgodnjo in stalno dostavo programske rešitve.
- Spremembe zahtev so dobrodošle tudi v poznih fazah projekta.
- Dostavljati delujoče verzije programa čim pogosteje.
- Vodstvo, uporabniki in razvojna ekipa mora sodelovati vsak dan.
- Projekt zagradimo okrog motiviranih posameznikov, ki jim nudimo zaupanje in podporo.
- Najboljši način prenosa informacij je pogovor v živo.
- Delujoča programska oprema je primarno merilo napredka.
- Pomembno je, da zagotovimo trajnostni razvoj.
- Stalno pozornost na tehnično odličnost in dobro oblikovanje krepi prilagodljivost.
- Preprostost – umetnost zmanjševanja nenarejenega dela, je bistvenega pomena.
- Razvojni inženirji si delo organizirajo sami.
- V rednih intervalih se mora razvojna skupina sestajati in predlagati, kako bi bila bolj učinkovita ter ustrezno odreagirati.

Ob upoštevanju zgornjih načel skuša vodja projekta in njegova ekipa čim prej naročniku zagotoviti rezultat. Trudijo se zagotoviti hitre in pogoste izdaje programske opreme. Na ta način lahko dobimo povratno informacijo uporabnikov programske rešitve in ustrezno ukrepamo. Tako lahko z naslednjo izdajo popravimo pomanjkljivosti, pripombe ali kakšne spremembe, ki so bile odkrite in posredovane s strani naročnika.

Eden od ciljev agilnega pristopa je okrepiti sodelovanje med ljudmi in poudariti, da je izmenjava informacij ključnega pomena. Agilnost je tudi sinonim za prilagodljivost. Agilni pristop cilja na sposobnost hitrega odziva projektne ekipe, na morebitne spremenjene uporabniške zahteve.

4.3.2 ZASTOPANOST AGILNEGA PRISTOPA V ORODJIH

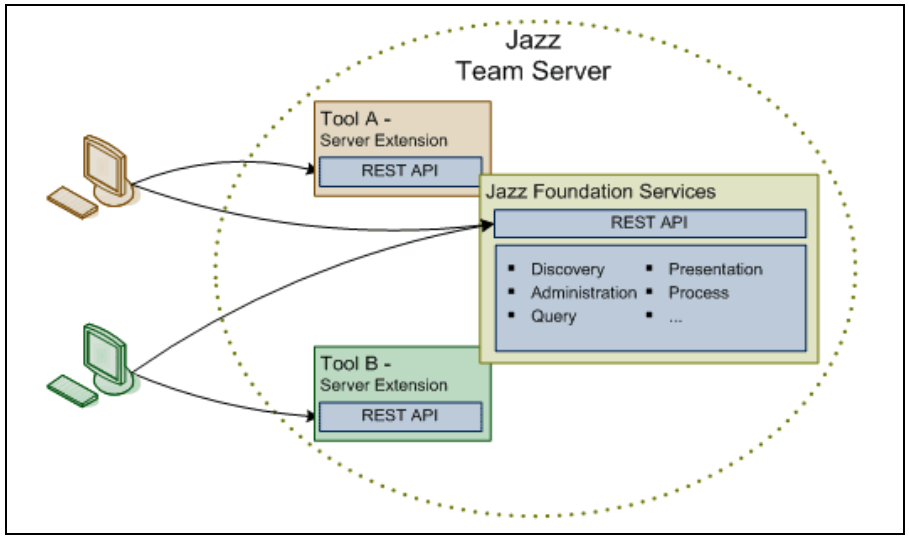
Razvijalci, ki razvijajo sodobna orodja, katera nudijo podporo pri vodenju in razvoju projektov IT se trudijo, da bi le-tega čim bolj podprla. Današnji način razvoja skuša vse bolj slediti agilnim tehnikam. Temu trendu sledijo tudi razvijalci orodij in skušajo skozi funkcionalnosti izdelkov podpreti agilne pristope. Vodilni ponudniki na tem področju že nekaj časa nudijo orodja, ki omogočajo vodenje projekta na agilen način. To pomeni, da morajo biti funkcionalnosti orodij zelo prilagodljive. Posamezne funkcionalnosti bi morale čim bolj slediti opisanim načelom iz poglavja 4.3.1. Zaželeno je tudi, da lahko uporabnik sam definira kakšna pravila v orodju in na ta način izrabi možnosti prilagajanja orodja.

4.4 IBM RATIONAL TEAM CONCERT

4.4.1 PLATFORMA JAZZ

IBM je ustvaril platformo JAZZ na podlagi želje, da bi razvojni skupini zagotovil učinkovito razvojno okolje. Eden izmed ciljev platforme JAZZ je zagotoviti visok nivo sodelovanja znotraj razvojne skupine, ki posledično dvigne tudi njeno produktivnost. Hoteli so poudariti, da je komunikacija znotraj skupine zelo pomembna in so jo zato postavili na prvo mesto. Drugi velik cilj je bil avtomatizirati čim več delovnih procesov, ki se pojavijo pri vodenju in razvoju. Tako lahko razvojna skupina prihrani čas, saj se ponovljivi procesi avtomatizirajo. S tem se tudi zmanjša možnost napak, ker je človeški faktor prisoten v manjši meri. Tretji večji cilj je bil zagotoviti dostop do ažurnih informacij, o stanju projekta v vsakem trenutku. Na voljo je veliko različnih izpisov, ki jih lahko uporabnik generira z nekaj kliki po uporabniškem vmesniku orodja RTC. Tako je zagotovljen dostop do aktualnih podatkov o stanju projekta v vsakem trenutku. To je velika pridobitev za vodjo projekta, saj njegove odločitve temeljijo na podlagi podatkov v realnem času.

Osrednji del arhitekturne zgradbe predstavlja strežnik Jazz Team Server (JTS). JTS nam nudi Jazz Foundation Services (JFS). JFS so osnovne storitve, ki omogočajo, da različna orodja delujejo skupaj. Tako lahko skupaj delujejo orodja za upravljanje z uporabniki, orodja za zagotavljanje varnosti, orodja za iskanje in še druga, ki temeljijo na JTS. Sama platforma JAZZ je zasnovana tako, da jo lahko enostavno razširimo z dodatnim orodjem, ki uporablja storitve JFS. Pri razširitvi je lahko na novo dodano orodje, tudi fizično na drugem strežniku. Komunikacija med strežniki pa poteka s pomočjo spletnega servisa. Arhitekturno zgradbo platforme JAZZ predstavlja slika 5.



Slika 5: Jazz Team Server

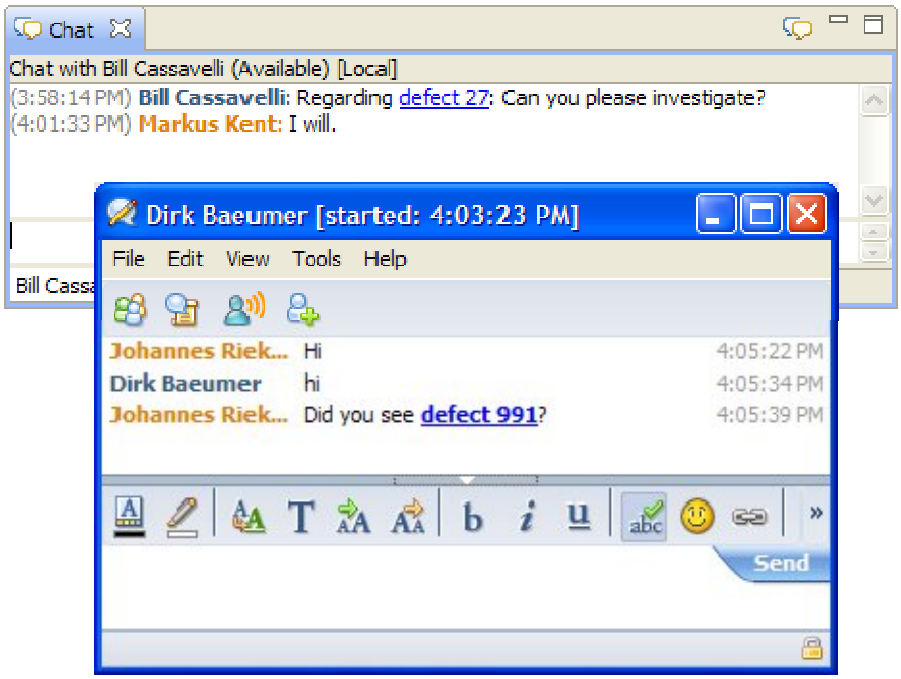
4.4.2 FUNKCIONALNOSTI RTC

RTC, produkt podjetja IBM, je eden izmed vodilnih primerkov računalniških orodij, ki se uporabljajo za pomoč pri vodenju in razvoju informacijskih rešitev. Na tak položaj ga uvrščajo njegove funkcionalne lastnosti in prednosti, ki jih nudi uporabniku.
Večino naštetih funkcionalnosti spodaj je opisanih v poglavju 4.2, zato bom v nadaljevanju izpostavil lastnosti, ki so specifične za RTC in tiste, ki še niso bile dovolj podrobno predstavljene.

Sistem za skupinsko uporabo

Pri zasnovi RTC je bilo veliko pozornosti posvečeno skupinski uporabi. To se tudi odrazi pri uporabi. Samo orodje je zelo prilagodljivo in omogoča, da si orodje konfiguriramo na način, ki je najbolj podoben procesu, ki že teče znotraj posamezne organizacije. Tako lahko za vsak projekt ustanovimo projektno okolje. V to okolje vključimo skupine, ki predstavljajo člane ljudi pri projektu, ki delajo na enakem področju. Primer skupine bi lahko bila skupina razvijalcev, skupina testerjev in druge. Na tak način lahko prenesemo že uveljavljeno strukturo relacij iz naše organizacije v RTC. Orodje omogoča tudi obveščanje vseh članov skupine o nastalih spremembah. To je hiter vir informacij od dogajanja v celotnem projektu do posameznika. Projektni vodja na ta način prihrani veliko časa, saj ne rabi vsakega posebej seznanjati z novostmi, ker so obvestila o spremembah, v večini primerov, generirana avtomatsko. Podlaga za avtomatsko kreiranje obvestil so lahko sprememba delovna naloge, uporabnika, skupine in drugi dogodki.

Zelo priročna zadeva je tudi klepetalnica, ki jo nudi RTC. Preko nje si lahko člani skupine hitro izmenjujejo informacije, pa tudi pošiljajo dokumente.

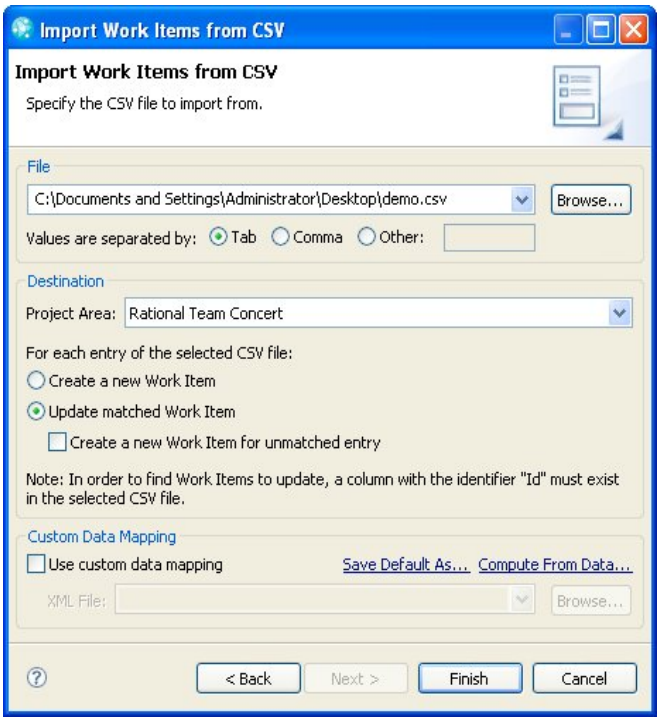


Slika 6: Sodelovanje: klepetalnica

Vodenje in sledenje delovnim nalogam

Mehanizem za upravljanje z delovnimi nalogami v RTC je zelo prilagodljiv in dodelan. Delovno nalogo lahko ustvarimo na podlagi predloge. Predlogo lahko uvozimo, namestimo svojo predlogo ali ustvarimo predlogo, iz že obstoječe delovne naloge. To velja tudi za atribute delovnih nalog. Uporabniku tudi omogoča, da sam določa, skozi katera stanja bo naloga prehajala v svojem življenjskem ciklu. Torej, stanja lahko določi projektni vodja in s tem uvede sistem dela, ki se mu zdi najboljši in najbolj primeren za določen projekt. Omogočeno je tudi povezovanje delovnih nalog. Vodja jih lahko poveže v hierarhijo od splošne do najbolj podrobne delovne naloge nekega vsebinskega sklopa.

RTC tudi omogoča dodajanje slik kot prilogo k delovni nalogi. V ta namen ima tudi vgrajeno posebno orodje, ki lahko posname sliko zaslona. Vsebino ene delovne naloge lahko tudi natisnemo na papir, prav tako lahko natisnemo več delovnih nalog naenkrat in dobimo natisnjen seznam v obliki tabele. Omogočen je tudi uvoz napak iz Bugzille in delovnih nalog iz datoteke, ki je formata z vejico ločenih vrednosti (Comma separated values – CSV). Če uvažamo iz formata CSV, lahko obstoječe delovne naloge tudi posodobimo.



Slika 7: Uvoz delovnih nalog iz datoteke

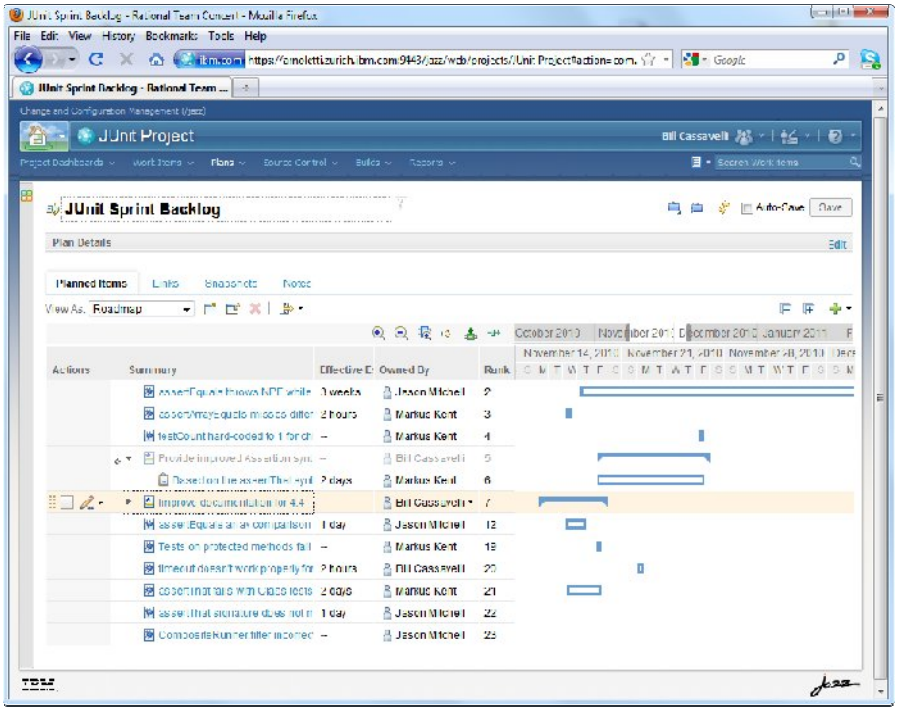
Iskanje delovnih nalog je zelo preprosto. Poizvedba je lahko tvorjena preko grafičnega uporabniškega vmesnika za posameznika. Poizvedbe lahko tudi delimo s celotno ekipo. RTC ima tudi vgrajen iskalnik potencialnih duplikatov delovnih nalog.

Sledenje spremembam posameznih nalog je omogočeno v preglednem izpisu, ki prikaže vse spremembe v kronološkem zaporedju. Uporabnik ima možnost nastavitve, da dobi obvestilo, kadar nastane sprememba na delovni nalogi.

Načrtovanje in vodenje projekta

Delovne naloge lahko vključimo v plan projekta. RTC nudi predloge za kreiranje plana po smernicah agilnega in tradicionalnega vodenja.
Pri tehniki agilnega vodenja ima projektni vodja na voljo v orodju tri osnovne predloge, ki mu pomagajo pri upravljanju in vodenju projekta. Prva je predloga za pregled seznama zahtev celotnega projekta (ang. Product Backlog), ki temelji na podlagi urejevalnika plana za celoten projekt. Druga je predloga za pregled seznama zahtev pri posamezni izdaji (ang. Release Backlog), ki tudi temelji na urejevalniku plana. Ta je še posebej primerna, kadar gre za velik in obsežen projekt z več izdajami. Predloge vodji nudijo enostavno premikanje načrtovanih enot plana med posamezne iteracije. Tretja je predloga za načrtovanja seznama nalog (ang. Planning a Sprint), ki nam omogoča pogled zgradbe seznama vse do individualne delovne naloge (ang. work breakdown structure) in urejanje, razporejanje delovnih nalog v sezname. Obenem imamo pregled nad uravnoteženostjo količine delovnih nalog med seznami.

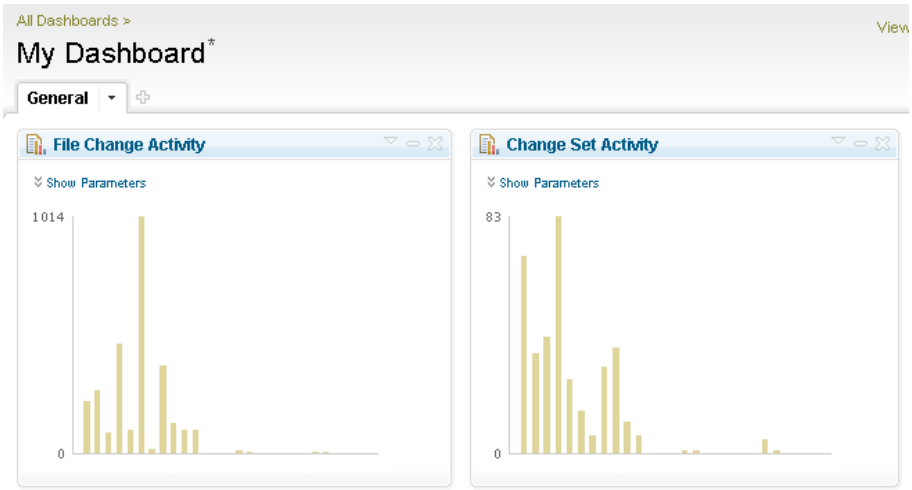
Vodja lahko v posebnem pogledu orodja načrtuje čas za seznam nalog, ki je potreben za izvedbo. RTC nam tudi prikaže stopnjo rizika, da bo seznam delovnih nalog pravočasno končan. Za to oceno mora vsak razvijalec oceniti svoje delovne naloge, s kakšno verjetnostjo jih bo dokončal v predvidenem času. Če to vsi naredijo, dobi projektni vodja na podlagi ocen vsakega razvijalca oceno, s kakšno verjetnostjo bo seznam delovnih nalog končan, glede na čas, ki ga je predvidel. Vsakemu razvijalcu je na voljo tudi najnižji nivo planiranja dnevnega dela skozi pogled »Moje delo«.
Če se vodi projekt na tradicionalni način, ima projektni vodja na voljo dve možnosti. Lahko planira celoten projekt hkrati, ali pa plan razdeli glede na faze projekta. Ne glede na to, kako se odloči, ima na voljo predlogo za tradicionalen način planiranja. Ko jo izpolni, ima na voljo pregled delovnih nalog in odvisnosti med njimi v Gantovem diagramu. Tu je tudi lepo prikazana kritična pot, ki prikazuje, pri katerih delovnih nalogah ne smemo zamujati, če hočemo izpeljati projekt v zastavljenem času.



Slika 8: Agilno načrtovanje

Upravljanje z izvorno kodo

Izvorna koda projekta se nahaja na strežniku in je fizično ločena od razvojne kode posameznih razvijalcev. Ko uporabnik hoče dodati svoje lokalne spremembe k izvorni kodi na strežniku, se najprej njegove spremembe shranijo na strežniškem privatnem prostoru, ki je namenjen razvijalcu. Eden razvijalec ima lahko več privatnih prostorov na strežniku. Nato se spremenjena koda iz privatnega strežniškega prostora doda izvorni kodi in tako deli z ostalimi uporabniki. Spremenjena koda se prenese v celoti, ali pa sploh ne. Na ta način je izvorna koda zaščitena pred napakami, ki bi se pojavile, če bi se prenesel samo del kode. Uporabnik ima možnost zaklepanja kode in se s tem izogne morebitnemu spajanju spremenjene kode z drugim uporabnikom če bi oba v istem trenutku spreminjala kodo na enakem delu. RTC ima implementiran tudi iskalnik, s katerim lahko uporabnik poišče spremembe, ki jih je naredil. Prav tako lahko uporabnik posamezno množico sprememb poveže delovno nalogo. Ob vsaki spremembi izvorne kode so uporabniki obveščeni. Za to poskrbi mehanizem za obveščanje. Mehanizem upravljanja z izvorno kodo nam omogoča tudi pregled, s kakšno aktivnostjo se spreminja izvorna koda. Vodja lahko na podlagi tega izpisa sklepa, ali se že bliža primeren trenutek za izdajo, saj pričakuje, da se aktivnost spreminjanja zmanjšuje, ko se projekt ali posamezna iteracija približuje koncu.



Slika 9: Izpis aktivnosti sprememb kode

Grajenje izvorne kode (ang. build)

Grajenje izvorne kode v RTC lahko poteka povsem avtomatsko. Na voljo je planer, v katerem lahko določimo, kdaj se požene naslednja iteracija grajenja kode. Nastavimo lahko tudi periodo časa, na podlagi katere se sproži grajenje. Uporabnik ima možnost zgraditi kodo tudi samo iz svoje kode. To doseže tako, da nastavi pod do kode, kjer je njegov privatni prostor na strežniku.

Grajenje lahko steče, če je bila dodana izvorni kodi vsaj ena množica sprememb. Ker so množice sprememb povezane z delovnimi nalogami, lahko hitro odkrijemo napako, če se pojavi pri graditvi. Uporabnik ima tudi možnost kreiranje nove delovne naloge na podlagi rezultata grajenja kode. Ob vsakem grajenju se posname koda, ki je bila uporabljena pri

grajenju. S tem je zagotovljeno, da lahko vedno, tudi kasneje, ustvarimo delovno okolje s kodo, ki je bila uporabljena pri grajenju.

Uporabnik lahko na podlagi izpisov, ki jih nudi RTC, primerja več grajenj med sabo. Primerja lahko stopnjo napak, trajanje grajenja in število sprememb, ki je zajeto pri novem grajenju.

Poročila in dokumentacija

Razna poročila in izpisi so na voljo za vse zgoraj naštete funkcionalnosti RTC. Pomen poročil je že opisan v poglavju 4.2.5, posamezni primeri poročil, ki jih nudi orodje so navedeni tudi v poglavju 4.4.2. Naj omenim še enkrat zelo pomembno dejstvo. Orodje omogoča kreiranje izpisov na podlagi trenutnega stanja projekta. Zato imamo pri roki vedno sveže informacije, ki so za vodjo projekta ključnega pomena. Na podlagi njih lahko sprejema odločitve, ki so za tisti trenutek pomembne in pravilne.

4.5 JIRA

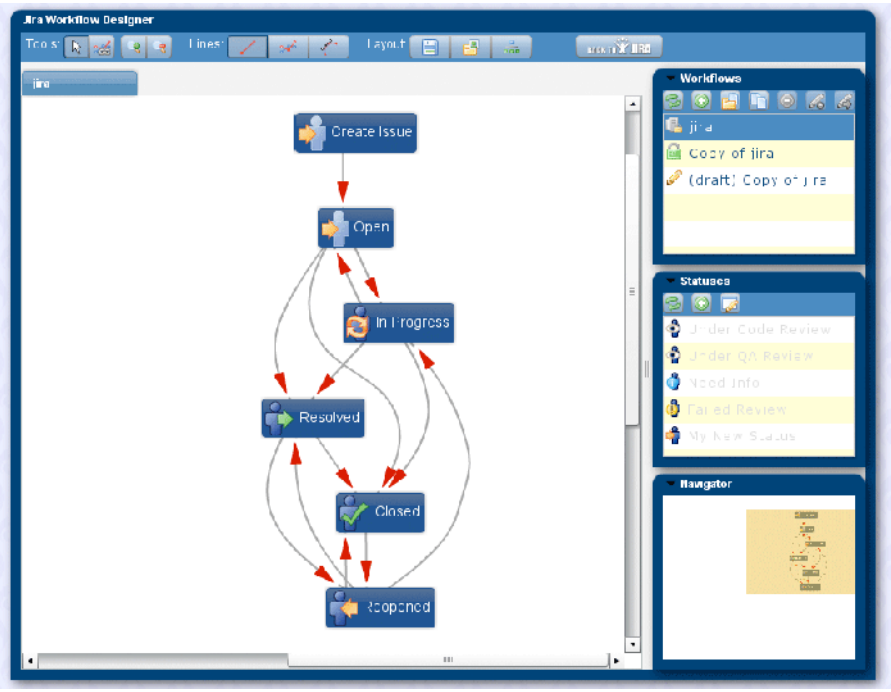
Jira je računalniško orodje, ki nudi podporo pri vodenju, načrtovanju in spremljanju stanja projektov. Najbogatejša različica zajema tudi funkcionalnosti, ki so namenjene za podporo razvoja IT projekta. Orodje je zelo razširjeno in sodi med skupino boljših orodij na tem področju. Orodje omogoča prilagodljivost uporabniku. To pomeni, da je zasnovan na način, da lahko uporabnik sam določa posamezna pravila in s tem karseda prilagodi uporabo orodja svojemu obstoječemu poslovnemu procesu.

Jira se ponaša z naslednjimi funkcionalnostmi:

- sledenjem in upravljanjem projekta,
- sledenjem in upravljanjem delovnih nalog,
- oglasno desko za objavo poročil in
- pripravo poročil in izpisov.

Kot sem že omenil, je Jira prilagodljivo orodje. Program omogoča definiranje različne vloge uporabnikov, ki sodelujejo na projektu, in kategoriziranje posamezne razvojne ekipe določenemu projektu. Na tak način dobi projektni vodja zelo dober pregled, katero vlogo zastopa posamezen član projekta, k kateri projektni skupini spada, na katerem projektu dela ta skupina in katere delovne naloge so dodeljene tej vlogi.

Uporabniku omogoča, da lahko sam predpiše, kako bo potekal tok delovnih nalog in skozi katera stanja bodo prehajale delovne naloge skozi svoj življenjski cikel. V okviru načrtovanja in urejanja delovnih nalog omogoča povezovanje le-teh v skupine in iteracije ter na ta način pripravo načrta za celotni projekt. Jira podpira tudi agilni način vodenja projekta.



Slika 10: Možna stanja delovne naloge

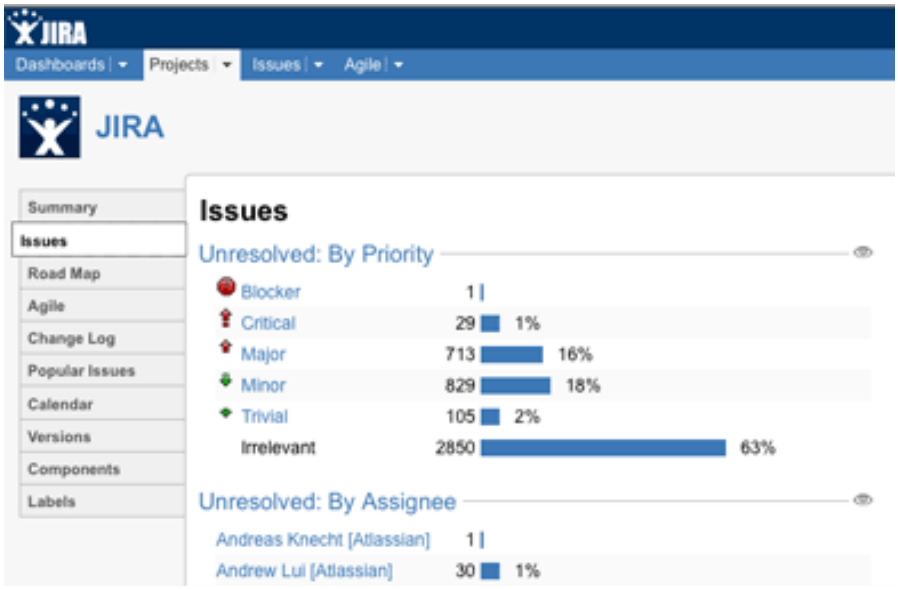
Lepa lastnost Jire je tudi obveščanje sodelujočih pri projektu, ko se zgodi sprememba na kakšni od delovnih nalog. Ker lahko ob prehodu delovne naloge iz enega stanja v drugo, nastavimo kakšen dogodek se proži, lahko zagotovimo informiranost celotne ekipe o nastali spremembi.

Skozi iskalnik, ki nam omogoča iskanje posameznih delovnih nalog, lahko sestavimo tudi kakšno poročilo o stanju projekta. To pa ni edini način, ki nam poda informacije o projektu. Jira nam omogoča izpis poročil in grafov, kot so: poročilo o dodeljenem delu razvijalcu, koliko dela je že opravljenega v primerjavi z načrtovanim obsegom, povprečen čas, ki je bil porabljen za realizacijo delovne naloge in druge izpise ter poročila.

Vse te lastnosti, ki jih ima Jira, lahko s pridom uporabi projektni vodja sebi v prid. Orodje mu nudi kakovosten pregled nad realnim stanjem projekta. To pa je ravno tisto, kar zanima vodjo, da lahko uspešno vodi projekt.

Jira nudi velik nabor dodatnih vstavkov. Ti se lahko vključijo oziroma povežejo s programskim orodjem Jira. Z dodatnimi funkcionalnostmi postane orodje res priročno in uporabno z vseh strani in nudi temeljito podporo vodenju in razvoju IT projekta. Nekateri od možnih vstavkov so:

- vstavek za upravljanje z izvorno kodo (Subversion),
- vstavek za podporo agilnega upravljanja s projektom (GreenHopper),
- vstavek za upravljanje grajenja kode (Bamboo),
- vstavek za upravljanje z razvojno kodo (FishEye) in drugi.



Slika 11: Prikaz nerešenih delovnih nalog po prioriteti

5. SKLEPNE UGOTOVITVE

Razvijanje novih programskih rešitev na področju IT predstavlja tipično projektno organizirano delovno okolje. V Sloveniji ni veliko podjetij s področja IT, ki bi imelo veliko število zaposlenih. Večina jih je manjših z nekaj 10 zaposlenimi. To pomeni, da poteka razvoj v veliki meri v manjših razvojnih skupinah. Tudi sam sem že pridobil nekaj izkušenj iz manjše razvojne skupine, ki ni uporabljala posebnega računalniškega orodja, ki se uporabljajo za pomoč pri vodenju in razvoju projekta. Med praktičnim delom sem opazil pomanjkljivosti, ki bi jih lahko omenjeno orodje odpravilo.

Podrobneje sem preučil orodje RTC. Seznanil sem se z njegovimi funkcionalnostmi. Moram priznati, da ima zelo širok nabor lastnosti, ki na enostaven in učinkovit način pomagajo pri vodenju projekta. Zelo dobro imajo podprto načrtovanje in razporejanje dela, ki ga je potrebno opraviti za projekt. Omogočajo sistematično urejanje in dodeljevanje posameznih delovnih nalog med člane projekta. Orodja tudi skrbijo za izredno natančno sledljivost vsem nastalim spremembam.

Vodji projekta omogočijo hiter in kakovosten pregled nad trenutnim stanjem projekta. To je v RTC realizirano z različnimi možnostmi kreiranja izpisov in poročil, ki jih pridobimo preko orodja, v nekaj klikih po uporabniškem vmesniku. Nepogrešljiv je tudi iskalnik, preko katerega lahko poiščemo posamezne ali skupine delovnih nalog. To so ključne informacije za vodjo, saj na podlagi teh sprejema nove odločitve. Lepota pa je tudi v tem, da so pridobljene hitro in predstavljajo realno stanje projekta. Vodja projekta pa ni edini, ki ima korist pri uporabi. Tudi vsi ostali člani projekta, ne glede na njihovo vlogo, preko implementiranih rešitev v RTC veliko pridobijo.

Prednost uporabe takega orodja se pokaže tudi v hitrosti prenosa posameznih informacij med člani projekta, saj snovalci teh orodij temu posvečajo zelo veliko pozornosti. Jira omogoča samodejno proženje obvestil, kadar nastane kakšna sprememba. Uporabnik pa lahko sam določi pravila za proženje teh obvestil. RTC ima vgrajeno tudi klepetalnico.

Kot slabost takšnih orodij bi izpostavil obvladovanje vseh različnih funkcionalnosti in možnosti, predvsem na začetku uporabe. Neizkušen uporabnik se lahko ob začetku uporabe malce zmede. V ta namen je na voljo dobra dokumentacija, ki pomaga začetnikom pri uporabi.

Uporabo orodja Jira bi priporočal manjšim razvojnim skupinam in njihovim projektnim vodjem. Prednosti Jire so predvsem v beleženju stanja projekta in v sledenju spremembam delovnih nalog na projektu. Razširjena različica programa pa tudi zadostno podpre sam razvoj IT rešitve. Prednost Jire pred RTC vidim v enostavnejši uporabi, predvsem za začetnika.

Orodje RTC je bolj kompleksno. To velja s stališča njegovih lastnosti in funkcionalnosti, teh je res veliko, kot tudi z vidika zahtevnosti uporabe. Menim, da je RTC primeren bolj za razvojne skupine, ki imajo nekaj več zaposlenih. Seveda pa to ne pomeni, da ga ne smejo uporabljati tudi manjše razvojne skupine. RTC s svojo funkcionalnostjo poskrbi za zelo dobro podporo vodenju in razvoju projekta. Prednost vidim tudi v tem, da za orodjem stoji velika

korporacija IBM, ki ga bo hitro nadgrajevala in skrbela za to, da se bo orodje razvijalo skupaj z najsodobnejšimi trendi na področju IT.

Funkcionalnosti omenjenih orodij so zares uporabne in mislim, da lahko pripomorejo k boljši organiziranosti, informiranosti in učinkovitejšemu sodelovanju članov projekta. Posledično se izboljšajo tudi rezultati in produktivnost ekipe. Žal nisem imel še možnosti, da bi lahko pozitivne učinke uporabe orodij preizkusil tudi v praksi. Vseeno sem pa prepričan, da manjša razvojna skupina z uporabo takega orodja pridobi več, kot izgubi.

6. VIRI

[1] Franc Solina, Projektno vodenje razvoja programske opreme, Ljubljana, 1997, pogl. 3.

[2] John C. Goodpasture, Project Management the Agile Way, Fort Lauderdale, 2010, pogl. 1

[3] Steve McConnell, Rapid Development, Redmond, Washington, 1996, pogl. 3.

[4] (2011) Spletno mesto EMRIS. Dostopna na: http://www2.gov.si/mju/emris.nsf

[5] (2011) Spletno mesto orodja Rational Team Concert. Dostopno na: http://jazz.net/projects/rational-team-concert

[6] (2011) Spletno mesto orodja Jira. Dostopno na: http://www.atlassian.com/software/jira