Univerza v Ljubljani

Fakulteta za računalništvo in informatiko

Igor Rožanc

Agilne metodologije razvoja programske opreme

Metoda SCRUM

Ekstremno programiranje (XP)

Razmerje med agilnim in discipliniranim pristopom

Študijsko gradivo za interno uporabo pri

predmetu Razvoj programskih sistemov 2

Ljubljana, 2008

Kazalo 2

Agilne metodologije: Uvod

Problemi plansko vodenih metodologij:

- Dolg življenski cikel
- Zahtevne za učenje in uporabo, težko prilagodljive
- Birokratske, preveč dokumentacije
- Težave pri vključevanju sprememb med razvojem
- Nove vrste PO zahtevajo drugačen, učinkovitejši razvoj

Karakteristike agilnega pristopa:

- Inkrementalen razvoj: kratki cikli, rezultat cikla je delujoča koda
- Intenzivno sodelovanje z uporabniki: stalna komunikacija
- Enostavnost metod: za učenje in uporabo
- Prilagodljivost metod:sposobnost sprotnega vključevanja sprememb zahtev naročnika

Agilne metodologije: Manifest agilnosti

Izhodišča:

- Iterativno izboljševanje kode progama (Basili,Turner 1975)
- Razvoj metodologij v 90. letih

Manifest agilnosti (Agile Manifesto):

- Srečanje leta 2001
- 17 svetovalcev in praktikov
- agilemaniphesto.org , www.agilealliance.org
- **Cilji manifesta:**
  - "nov" način razvoja programske opreme
  - ne proti, pač pa za ponovno uporabo metodologij
  - modeliranje in dokumentiranje v smiselnem obsegu

Agilne metodologije: Manifest agilnosti

Agilne metodologije poudarjajo:

- Ljudi na projektu in komunikacijo med njimi
  pred
  procesom in razvojnimi orodji
- Hitro razvit, stestiran in delujoč program
  pred
  popolno dokumentacijo brez delujočega programa
- Komunikacijo in sodelovanje med razvijalci in naročnikom
  pred
  pogodbeno dogovorjenem razvojem PO
- Prožnost pri spremembah zahtev
  pred
  sledenjem začrtanemu načrtu pri razvoju PO

Agilne metodologije: Manifest agilnosti

12 osnovnih principov agilnih metodologij:

1. Najvišja prioriteta je zadovoljiti naročnika s hitro in neprestano dostavo uporabne PO.
2. Spremembe zahtev se spodbujajo, tudi pozneje v razvoju. Agilni proces šteje upoštevanje sprememb zahtev kot primerjalno prednost za naročnika.
3. Dostava delujoče PO naj bo čim pogostejša, od nekaj tednov (zaželjeno) do največ nekaj mesecev.
4. Naročniki in razvijalci naj pri razvoju vsakodnevno sodelujejo.
5. Projekte je treba zastaviti okrog motiviranih posameznikov. Tem je treba nuditi ustrezno okolje in potrebne vire, predvsem pa jim je treba zaupati, da bodo zastavljeno delo opravili.
6. Najprimernejši in najučinkovitejši način za posredovanje informacij o in za projekt je osebni pogovor.

Agilne metodologije: Manifest agilnosti

12 osnovnih principov agilnih metodologij (nadaljevanje):

7. Najpomembnejše merilo napredka je količina delujoče programske kode.
8. Agilni procesi predvidevajo nenehni razvoj v okviru projekta. Naročniki, razvijalci in sponzorji morajo biti pripravljeni stalno in na ustrezen način podpirati razvoj v okviru projekta.
9. Nenehna težnja za tehnično odličnost in optimalen načrt razvijajoče PO pospešuje agilnost.
10. Bistevega pomena je enostavnost – sposobnost maksimiziranja tistega dela razvoja, ki še ni razvit.
11. Najboljša opredelitev zahtev, načrti in programska koda nastane pri skupinah, ki se samostojno organizirajo in vodijo.
12. Občasno (ampak redno) mora skupina preverjati, kako bi postala učinkovitejša, in v ta namen sprejeti ustrezne ukrepe.

Agilne metodologije: Seznam

Seznam obstoječih agilnih metodologij:

a) Agilna metodologija Scrum
b) Ekstremno programiranje – XP (ang. eXtreme Programming)
c) Agilne tehnike podatkovnih baz – AD (ang. Agile Database Techniques)
d) Agilno modeliranje – AM (ang. Agile Modeling)
e) Prilagodljiv razvoj PO – ASD (ang. Adaptive Software Development)
f) Družina metodologij Crystal
g) Funkcijsko voden razvoj PO – FDD (ang. Feature Driven Development)
h) Metoda dinamičnega razvoja sistemov – DSDM (ang. Dynamic Systems Development Method)
i) "Vitek" razvoj PO – LSD (ang. Lean Software Development)
j) Testno voden razvoj – TDD (ang. Test-Driven Design)

Agilne metodologije: SCRUM

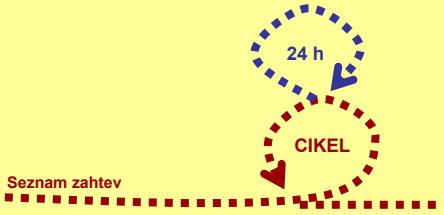
Izkustven razvoj PO s poudarkom na prožnosti, prilagodljivosti in ustvarjalnosti

- Izvor imena: Rugby - vrnitev žoge v igro
  - pomen dela skupine in predvsem samoorganizacije
- Pokriva vodenje projekta, manj natančno proces razvoja in tipične prakse, aktivnosti in izdelke
- Razlika med definiranimi in empiričnimi procesi - drugačen način vodenja:
  - empiričen proces ni enostavno ponovljiv
  - zahteva stalen vpogled in prilagajanje
- Naloga vodstva:
  - NE predpisovati, kaj mora skupina delati
  - ustvarjati okoliščine, ki čimmanj motijo delo razvojne skupine.
- Skupine se za delo organizirajo same

Agilne metodologije: SCRUM

Ogrodje metodologije: iterativen inkrementalen proces

- Razvoj poteka v iteracijah, ki si sledijo ena za drugo
- Rezultat vsake iteracije je nova (dodatna) funkcionalnost izdelka
- Pomembni so vsakodnevni pregledi znotraj enega cikla
- Ti omogočajo članom skupine, da medsebojno pregledajo opravljeno delo in izvedejo prilagoditve
- Osnova za cikel je seznam zahtev – ko so vse izvedene, je cikel končan



Agilne metodologije: SCRUM

Vloge:

- Lastnik izdelka (ang. Product Owner)
- Razvojna skupina (ang. Team)
- Skrbnik metodologije (ang. Scrum master)
- Opazovalci (ang. Observers)

Lastnik izdelka (ang. Product Owner):

- Predstavlja vse, ki so zainteresirani za projekt in njegove rezultate
- Skrbi za zagotavljanje sredstev in upravičenost projekta
- Vzdržuje seznam zahtev (ang. Product Backlog)
  - izdela začetni seznam
  - določa prioriteto posameznih zahtev
  - planira predajo posameznih verzij izdelka v obratovanje

Agilne metodologije: SCRUM

Razvojna skupina (ang. Team):

- Odgovarja za razvoj novih funkcij znotraj vsake iteracije
- Deluje po principu samoorganizacije
  - člani sami izberejo način, kako realizirati posamezne zahteve
  - člani so sami odgovorni za uspeh posameznih iteracij in projekta v celoti

Skrbnik metodologije (ang. Scrum master):

- Odgovarja za skladnost celotnega procesa z metodologijo Scrum
  - uči metodologijo vse, ki sodelujejo pri projektu
  - skrbi, da se metodologija vklaplja v kulturo organizacije
  - skrbi, da metedologija daje pričakovane rezultate
  - zagotavlja, da se upoštevajo pravila in praksa metedologije

Agilne metodologije: SCRUM

Opazovalci (ang. Observers):

- Metodologija strogo loči med tistimi, ki so odgovorni za realizacijo projekta, in ostalimi zainteresiranimi opazovalci
- Opazovalci se ne smejo neposredno vmešavati
- Prašiči in piščanci ...

Potek projekta po metodologiji Scrum:

- vizija (ang. Vision)
- oblikovanje seznama zahtev (ang. Product Backlog)
- več tekov (ang. Sprint)
- sestanki (ang. Meetings)
- izdelki (ang. Products)

Agilne metodologije: SCRUM

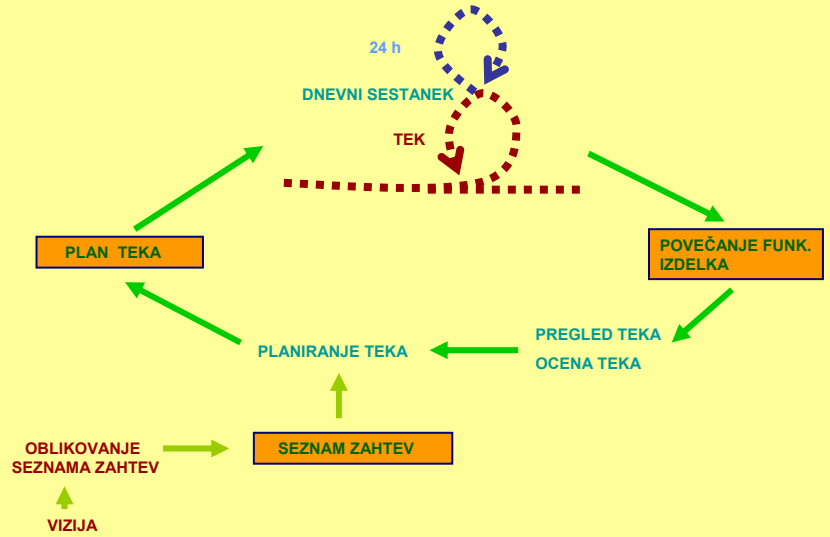
Sestanki (ang. Meetings):

- planiranje teka (ang. Sprint Planning Meeting)
- dnevni sestanek (ang. Daily Scrum)
- pregled teka (ang. Sprint Review Meeting)
- ocena teka (ang. Sprint Retrospective Meeting)

Izdelki metodologije Scrum:

- seznam zahtev (ang. Product Backlog)
- plan teka (ang. Sprint Backlog)
- povečevanje funkcionalnosti izdelka (ang. Increment of Potentialy Shippable Product Functionality)

Agilne metodologije: SCRUM



Agilne metodologije: SCRUM

Vizija (ang. Vision):

- Projekt se začne z vizijo, ki je lahko na začetku še nejasna
- Lastnik projekta je zadolžen za realizacijo vizije na način, ki v največji možni meri povrne vložena sredstva
- Vizija predvidi tudi okvirno število izdaj in mejnike

Oblikovanje seznama zahtev (ang. Product Backlog):

- Oblikovanje seznama funkcionalnih in nefunkcionalnih zahtev, ki (izvedene) uresničujejo zastavljeno vizijo
- Zahteve so razvrščene po prioriteti in grupirane v zaporedne izdaje
- Zahteve za začetne izdaje so podrobne, za kasnejše bolj grobe

Več tekov (ang. Sprint):

- Razvoj poteka v več iteracijah – tekih
- Vsak tek traja (okvirno) 30 dni

Agilne metodologije: SCRUM

Planiranje teka (ang. Sprint Planning Meeting):

- Izvaja se nazačetku vsakega teka
- Lastnik izdelka in razvojna skupina dogovorijo, katere zahteve bodo realizirali v tem teku
- Traja 8 ur in je sestavljen iz 2 delov po 4 ure:
  - najprej izberejo zahteve, ki jih bodo realizirali v tem teku
  - nato razvojna skupina izdela plan teka (Sprint Backlog)

Dnevni sestanek (ang. Daily Scrum):

- Vsakodnevni sestanek vseh odgovornih članov skupine
- Vedno ob isti uri, traja 15 do 30 minut
- Vsak član skupine odgovori na 3 vprašanja:
  - Kaj si delal od prejšnjega sestanka?
  - Kaj boš delal do naslednjega sestanka?
  - S katerimi težavami se srečuješ pri delu?

Agilne metodologije: SCRUM

Pregled teka (ang. Sprint Review Meeting):

- Izvaja se na koncu vsakega teka, traja največ 4 ure
- Razvojna skupina predstavi rezultate teka lastniku izdelka in ostalim zainteresiranim (opazovalcem)
- Lastnik izdelka in opazovalci podajo svoje pripombe in želje za realizacijo v naslednjem teku

Ocena teka (ang. Sprint Retrospective Meeting):

- Izvaja se na koncu vsakega teka, traja največ 3 ure
- Skrbnik metodologije in razvojna skupina ocenijo potek teka in predlagajo morebitne izboljšave procesa

Vsi sestanki omogočajo:

- sprotno pregledovanje procesa (inspection)
- sprotno prilagajanje procesa (adoptation)

Agilne metodologije: SCRUM

Seznam zahtev (ang. Product Backlog):

- Seznam ni nikoli dokončen:
  - na začetku projekta predstavlja začetno oceno zahtev
  - med delom se dopolnjuje in postaja vse bolj podroben
  - obstaja toliko časa, dokler je izdelek v uporabi
- Vrstice predstavljajo posamezne zahteve:
  - zahteve so grupirane v teke, teki pa v izdaje
  - zahteva, ki ni bila realizirana v enem teku, se prenese v enega od naslednjih tekov (odvisno od zahteve)
  - zahteve se stalno dodajajo
- Stolpci opisujejo karakteristike zahteve (v teku):
  - prvi 4 stolpci za opis zahteve: ime zahteve, začetna ocena v dnevih, faktor zahtevnosti, ocena z upoštevanjem faktorja zahtevnosti
  - ostali stolpci so vezani na posamezne teke:opis obsega dela, ki še ni bilo opravljeno; omogoča prikaz, koliko še manjka do konca projekta

Agilne metodologije: SCRUM

Ime zahteve	Ocena dela (dni)	Zah. faktor	Poprav. ocena	Čas potreben za dokončanje			
				1	2	3	4
Izdelava prijavne strani	4	0	4	4	0		
Določitev varnostne politike	8	0	8	8	0		
Funkcija #2	5	0	5	5	0		
Funkcija #3	7	0	7	7	0		
Tek 1	24	0	24	24	0		
Iskalnik po strani	4	0.1	4.4	4.4	4.4		
Funkcija #1	13	0.1	14.3	14.3	14.3		
Dopolnitev funkcije #2 s stranjo za pomoč	1	0.1	1.1		1.1		
Tek 2	18	0.1	19.8	18.7	19.8		
Izpis podatkov funkcije #2 v različnih formatih	6	0	6	6	6		
Verzija 1.0				48.7	25.8		
Funkcija #4	15	0.2	18	18	18		

Agilne metodologije: SCRUM

Plan teka (ang. Sprint Backlog):

- Predstavlja seznam nalog, ki morajo biti opravljene v enem teku
- Obseg dela za vsako nalogo mora znašati med 4 in 16 delovnih ur
- Grobo definirane ali preveč obsežne naloge je treba razbiti na več manjših
- Plan teka lahko spreminja samo delovna skupina
- Stolpci predstavljajo zaporedje posameznih delovnih dnevov v teku
- Za vsako nalogo vpisujemo obseg dela, ki še ni bilo opravljeno

Agilne metodologije: SCRUM

Opis naloge	Tvorec	Odgovorni	Status	Dnevi izvajanja teka					
				1	2	3	4	5	6
Izdelava prijavne strani									
- Mehanizem preverjanja gesla		Lojze	končano	10	4	0	0		
- Prebiranje podatkov certifikata		Martin	končano	16	8	1	0		
- Izdelava mehanizma za klice posameznih funkcij		Miha	v delu	25	18	10	6		
Določitev varnostne politike									
- Določitev tabel za hranjenje uporabnikov		Martin	končano	6	6	6	0		
- Določitev in implementacija varnostnih pravil		Martin, Miha		80	80	80	80		
- Preverjanje nivoja varnosti		Lojze		16	16	16	16		
Funkcija #2									
- Določitev potrebnih tabel		Lojze	končano	5	5	0	0		
- Računanje rezultata				8	8	8	8		

Agilne metodologije: SCRUM

Povečevanje funkcionalnosti izdelka (ang. Increment of Potentialy Shippable Product Functionality)

- Rezultat vsakega teka je dodatna funkcionalnost, ki je neposredno uporabna
- Rezultati morajo biti v taki obliki, da jih lahko takoj damo v uporabo
- Izdelek obsega:
  - dobro strukturirano, popolno napisano in v celoti stestirano programsko kodo
  - dokumentacijo v obliki navodil za uporabnika ali vgrajenih datotek za pomoč uporabniku
- Pri zahtevnih aplikacijah je potrebna še posebna odobritev oziroma pridobitev posebnega certifikata
  - v tem primeru je treba pripraviti še dodatno dokumentacijo
  - izdelek je v celoti končan, ko je pripravljen za odobritev

Agilne metodologije: SCRUM

Tipični primeri uporabe metode Scrum:

- majhen projekt, majhno število razvijalcev, potreba po hitrem rezultatu
- projekt z znanim okoljem in tehnologijo, a spreminjajočimi zahtevami
- projekt, ki raziskuje možnosti uporabe novega okolja in/ali tehnologije

Pozitivne izkušnje:

- boljše poznavanje trenutnega stanja projekta
- boljše sodelovanje med razvijalci, večji občutek pripadnosti
- višja delovna vnema
- manjša možnost in občutek frustracije ob neuspehih

Negativne izkušnje:

- nujno poznavanje in ustrezna uporaba metodologije
- problematična izbira vlog, učinkovito izvajanje določnosti in pravic
- problem nadzora nad celoto
- problem zapletenih in obsežnih projektov

Agilne metodologije: SCRUM

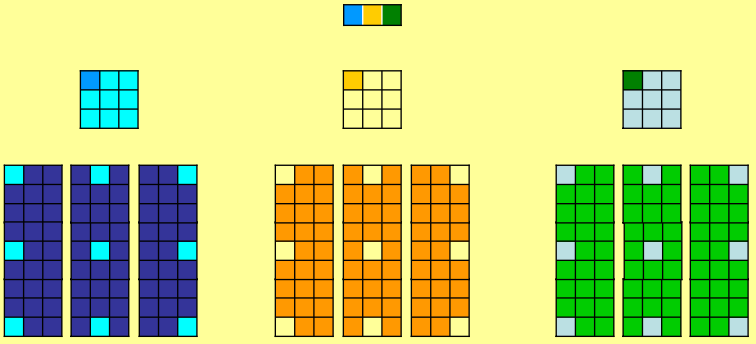
Prednosti uporabe SCRUM-a (po reviji IEEE Software):

- izdelava izdelka se izvaja z zaporedjem obvladljivih ciklov
- očiten je napredek, čeprav se zahteve sporeminjajo
- vsi udeleženci so seznanjeni z vsemi značilnostmi razvoja
- izboljša se medsebojna komunikacija v razvojni skupini
- vsa razvojna skupina je udeležena pri dosežkih med in po koncu razvoja
- naročniki vidijo sprotno naraščanje funkcionalnosti končnega izdelka
- naročniki sproti dobivajo informacije o realni kakovosti razvijajočega izdelka
- odnosi z naročniki se izboljšajo, razvojna skupina se poveže, znanje naraste
- ustvari se vtis, ko vsi udeleženci aktivno sodelujejo in pričakujejo uspeh razvoja

Agilne metodologije: SCRUM

Uporaba metodologije za večje skupine razvijalcev

- vsaka celica predstavlja enega razvijalca
- vsaka razvojna skupina delegira enega člana za Scrum na višjem nivoju



Agilne metodologije: Ekstremno programiranje

Ekstremno programiranje – eXtreme programming (XP):

- učinkovit, nizko tvegan, prilagodljiv, sistematičen in zabaven način analize, načrtovanja in izdelave programske opreme
- nobenega zares novega principa, le združitev nekaterih že znanih principov in njihova uporaba do skrajnosti (v ekstrem).

Izhodišča:

- spiralni razvoj (Boehm, 1998): postopen razvoj, z manjšimi dodatki
- XP (Kent Back, Ward Cunningham, Ron Jefries): učinkovito odzivanje na spremembe + kontrola virov

Štiri vrednote XP:

- komunikacija
- enostavnost
- povratne informacije
- pogum

Agilne metodologije: Ekstremno programiranje

XP nadomešča:

- kratki cikli → iterativni razvoj, vsebino iteracije določa igra planiranja
- arhitektura → stalno načrtovanje in izpopolnjevanje arhitekture z metaforami
- enostavnost → najenostavnejši načrt, ki podpira trenutno funkcionalnost
- načrtovanje → sprotno preoblikovanje kode
- pregledovanje kode → programiranje v parih
- testiranje → vnaprejšnja priprava testov in sprotno testiranje
- integration testing → neprestano sprotno vključevanje novosti

Prednosti XP

- vse prakse so smiselna celota
- prakse se med seboj dopolnjujejo
- zahteva se temeljito izvajanje praks

Agilne metodologije: Ekstremno programiranje

Šest faz življenskega cikla:

1) Raziskovalne faze (angl. exploration phase):

- naročniki napišejo kartice z zgodbami (angl. story card)
- ena kartica - ena funkcionalnost
- razvojna skupina spoznava tehnologijo, orodja, način razvoja
- (lahko) prototipi delovanja sistema

2) Faza načrtovanja (angl. planning phase):

- prednostna lista razvoja zgodbic
- vsebina prve verzije programa
- programerji določijo oceno dela in urnik razvoja
- urnik se določi za največ 2 meseca naprej

Agilne metodologije: Ekstremno programiranje

Šest faz življenskega cikla (nadaljevanje):

3) Faza ponavljanja do zaključka (angl. iteration to release phase):

- več ponavljanj v sklopu prve verzije programa
- urnik razvoja se razgradi na posamezna ponavljanja (1-4 tedne)
- naročnik določa vsebino razvoja na začetku vsakega ponavljanja
- naročnik preveri delovanje programa na koncu ponavljanja – požene funkcionalne teste
- na koncu zadnjega ponavljanja imamo delujoč program

4) Faza delovanja (angl. productionizing phase):

- pred predajo programa v uporabo opravimo dodatna testiranja pravilnosti in učinkovitosti delovanja programa
- lahko potreba po popravkih oz. dopolnitvah - rešujemo s (tipično krajšimi) ponavljanji (cca 1 teden)

Agilne metodologije: Ekstremno programiranje

Šest faz življenskega cikla (nadaljevanje):

5) Faza vzdrževanja (angl. maintainance phase):

- po prenosu prve verzije v delovanje
- poleg novega razvoja (naslednje verzije) vzdržujemo obstoječi programa, zato počasnejši razvoj

6) Faza umiranja (angl. death phase):

- ko naročnik nima več zgodbic
- (lahko) začnemo pisanje dokumentacije, ki se ne bo več spremenila
- ta faza lahko nastopi tudi preden vključimo vse zgodbice naročnika (neučinkovitost programa, predrag nadaljni razvoj, itd.).

Agilne metodologije: Ekstremno programiranje

14 XP principov:

a) Skupni načrt (angl. planning game):

- sodelujejo programerji in naročnik(i) s ciljem izdelati optimalen načrt
- naročnik napiše zgodbico
- zanjo programer oceni delo, potrebne vire ter medsebojno odvisnost z ostalimi zgodbicami
- na podlagi ocene naročnik določi, kdaj je potrebno zgodbico razviti (ponavljanje, verzijo)
- skupna odgovornost, integracija ostalih principov ...

b) Majhne in časovno kratke verzije (angl. small/short releases)

- XP primeren predvsem za enostavne sisteme (2-3 mesece razvoj)
- želimo čimkrajši razmak med verzijami (nekaj dni, največ 1 mesec)

Agilne metodologije: Ekstremno programiranje

14 XP principov (nadaljevanje):

c) Podoba (angl. metaphor)

- sistem definiran skozi podobe (predstave), ki se pojavljajo med naročnikom in programerji
- razvoj je podrejen temu, da se doseže vse podobe oz. glavno podobo programa

d) Preprost model (angl. simple design)

- pri razvoju programa uporabljamo čim enostavnejši model v danem trenutku
- takoj brišemo vse odvečnosti (nepotrebno kompliciranje, odvečna koda)

Agilne metodologije: Ekstremno programiranje

14 XP principov (nadaljevanje):

e) Avtomatizirano testiranje (angl. testing)

- XP zahteva ti. testno vodeno programiranje (avtomatizirani testi)
- najprej pripravimo testne razrede, nato kodiramo module programa
- testni razredi se med razvojem neprestano (in povsod) poganjajo
- (dodatno) naročnik pripravi funkcionalne teste

f) Preoblikovanje (angl. refactoring)

- nanehno preoblikovanje kode
- vsakič in takoj, ko programer med razvojem opazi:
  - ponavljanje kode
  - možnost poenostavitve
  - možnost optimizacije in podobno

Agilne metodologije: Ekstremno programiranje

14 XP principov (nadaljevanje):

g) Programiranje v parih (angl. pair programming)

- dva programerja pišeta kodo skupaj za enim računalnikom
- več možnih scenarijev: ekspert, povprečnež, začetnik, ekstrovertiranec, introvertiranec, ...
- nekoliko več časa, a se izplača zaradi kakovosti kode; ključno: učenje
- + : pozitivno rivalstvo, usklajevanja, skupno pregledovanje kode in odpravljanje napak, delitev znanja, pogum, zaupanje...
- - : odvisnost, izguba časa –podvajanje virov, tradicionalno individualna aktivnost, oseben odnos do kode, hrup, koncentracija, domišljavost, neusklajenost...
- Raziskava na University of Utah: ekonomičnost, reševanje problemov, zadovoljstvo, programerjev, učenje, kakovost načrtovanja, skupinsko delo, sprotno pregledovanje kode, vodenje ljudi in projekta

Agilne metodologije: Ekstremno programiranje

14 XP principov (nadaljevanje):

h) Skupno lastništvo (angl. collective ownership)

- skupna last kode - kdorkoli lahko spremeni katerikoli del programa
- zahteva jasne principe, enostavno strukturo izdelkov, znanje vseh razvijalcev, veliko komunikacije
- individualno lastništvo kode je ovira, saj se koda ne razvija tako hitro, kot bi se morala, ker drugi nočejo motiti lastnika
- vsak razvijalec je dolžan popraviti kodo, če vidi možnost izboljšave
- sprotna integracija preprečuje anarhijo
- že pripravljeni testi zagotavljajo, da v popravljeni kodi ni napak

Agilne metodologije: Ekstremno programiranje

14 XP principov (nadaljevanje):

i) Nepretrgana integracija (angl. continuous integration)

- vsak nov del kode se uporabi takoj, ko je pripravljen (testiran)
- najnovejša različica programa spremeni večkrat (2-3) na dan
- zahteva: ob vsakem vstavljanju nove kode se poženejo vsi obstoječi testi

j) 40 urni delavnik (angl. 40-hour week)

- dovolj, a ne preveč delovnega časa
- dovoljena sta kvečjemu dva zaporedna tedna s podaljšanim delovnim časom
- v nasprotnem primeru problem planiranja

Agilne metodologije: Ekstremno programiranje

14 XP principov (nadaljevanje):

k) Stalno dosegljiv naročnik (angl. on–site customer)

- XP zahteva, da je naročnik vedno dosegljiv razvojni ekipi za morebitna vprašanja in komentarje
- naročnik ima dolžnosti pri testiranju in prevzemu programov

l) Vzpostavljeni kodirni standardi (angl. coding standards)

- obstajajo standardna pravila pisanja kode
- vsi programerji se jih morajo držati
- to vodi v izboljšanje komunikacije in večjo kakovost

14 XP principov (nadaljevanje):

m) Primeren prostor (angl. open workspace)

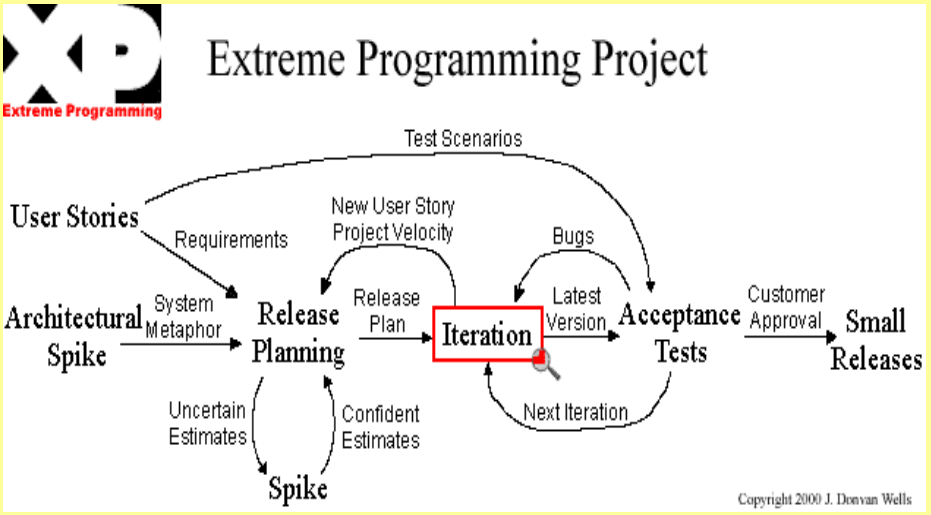
- za programerje velik prostor pregrajen na manjše prostore - celice (angl. cubicles)
- prednost – boljše sodelovanje in komunikacija

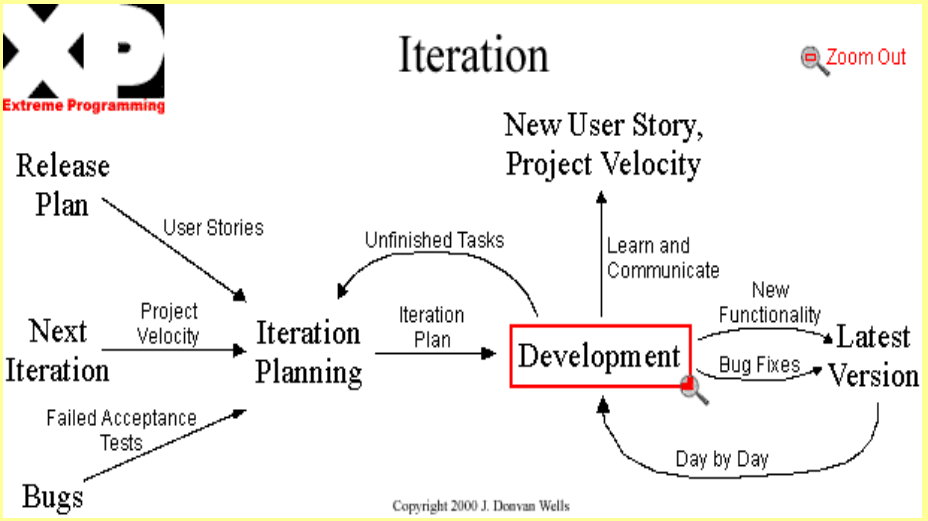
n) Korektna pravila (angl. just rules)

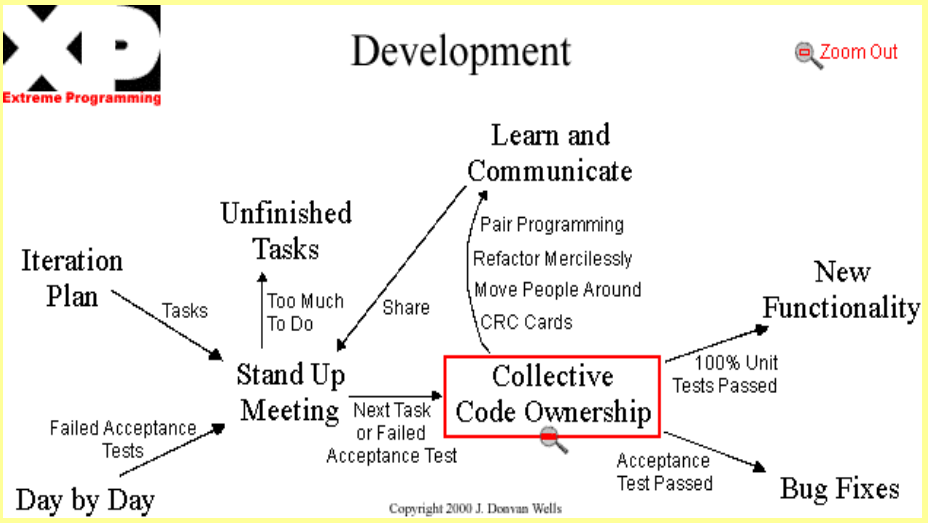
- razvojna ekipa si sama vzpostavlja pravila
- pravila so obvezna
- za spremembo pravil je potrebno soglasje in ocena prednosti

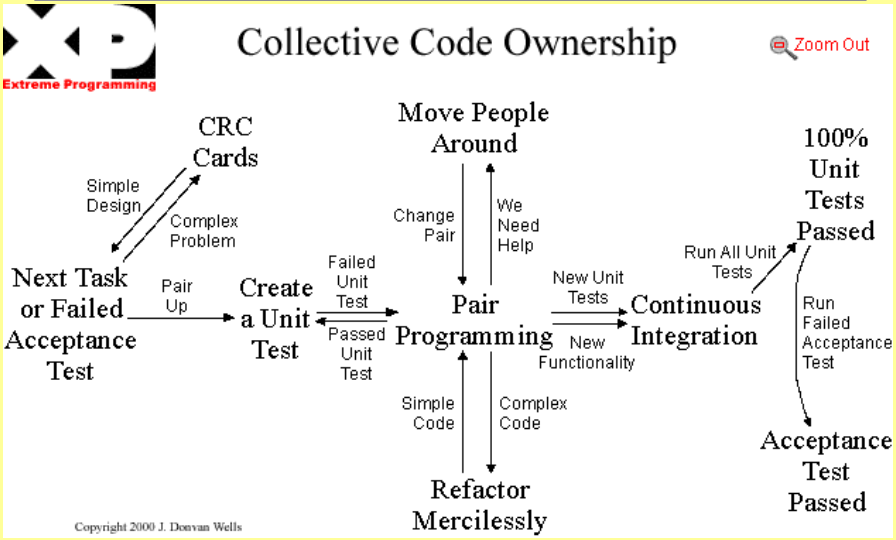
Pravilo 20 - 80

- **80 % koristi izhaja iz 20 % dela:** realiziraj 20 % funkcionalnosti, opravi 20 % najpomembnejšega načrtovanja
- **80 % praks prinaša samo 20 % koristi:** sinergičen učinek vseh praks




Iteration
Zoom Out
Extreme Programming
New User Story,
Project Velocity
Release
Plan
User Stories
Unfinished Tasks
Learn and
Communicate
New
Functionality
Next
Iteration
Project
Velocity
Iteration
Planning
Iteration
Plan
Development
Latest
Version
Bug Fixes
Failed Acceptance
Tests
Day by Day
Bugs
Copyright 2000 J. Donvan Wells


Development
Zoom Out
Extreme Programming
Learn and
Communicate
Unfinished
Tasks
Pair Programming
Refactor Mercilessly
Move People Around
CRC Cards
Iteration
Plan
Tasks
Too Much
To Do
Share
New
Functionality
Stand Up
Meeting
Next Task
or Failed
Acceptance Test
Collective
Code Ownership
100% Unit
Tests Passed
Failed Acceptance
Tests
Acceptance
Test Passed
Day by Day
Bug Fixes
Copyright 2000 J. Donvan Wells



Vsak princip ima pomanjkljivosti, ki jih rešujejo drugi principi, recimo:

a) Programiranje v parih

- kodirni standardi preprečujejo drobne prepire
- 40-urni delovni teden zagotavlja počitek in zmanjšuje verjetnost nekoristnih razprav
- vnaprejšnja priprava testov poveča razumevanje problema pred kodiranjem
- metafora služi kot vodilo pri načrtovanju
- enostaven načrt zagotavlja razumljivost

b) Preoblikovanje

- skupno lastništvo kode omogoča izvajanje vseh potrebnih sprememb
- kodirni standard zagotavlja večjo razumljivost kode
- programiranje v parih daje več poguma za preoblikovanje
- enostaven načrt olajša spremembo strukture
- sprotno testiranje zagotavlja, da tudi pravilno delovanje po preoblikovanju
- sprotna vključevanje novosti pokaže, če je preoblikovanje povzročilo težave ostalim
- 40-urni delovni teden zagotavlja, da so počitek in zmanjšuje verjetnost napak

Agilne metodologije: Ekstremno programiranje

Uporaba ekstremnega programiranja ni primerna:

- preveliki, predolgi ali preveč zahtevni projekti
- ko ga razvijalci niso pripravljeni sprejeti
- prekomerno in prepogosto prekoračen urnik 40 ur dela na teden
- uporaba določene tehnologije, ki eksponentno viša stroške
- prepočasno prihajajo povratne informacije od naročnikov
- nimamo pravilnega delovnega okolja
- obdobja med ponovitvami so daljša od 5 mesecev

XP je primeren za majhne sisteme, kjer se zahteve spreminjajo in hitro potrebujemo učinkovite rešitve!

XP uvajamo postopoma, najprej principe, ki so povezani z vodenjem projekta, izmed ostalih pa najprej testiranje!

Agilne metodologije: Pregled

Pregled ostalih agilnih metodologij in njihovih glavnih idej:

a) Agilne tehnike podatkovnih baz – AD (ang. Agile Database Techniques)

- samo razvoj podatkovnega aspekta PO
- podatki so najpomembnejši vidik, razvijamo jih postopoma (ne na začetku)
- izogibanje ekstremnim rešitvam, iskanje kompromisov
- unikaten pristop za vsak projekt
- sodelovanje vseh razvijalcev, vloge razvijalcev PB

b) Agilno modeliranje – AM (ang. Agile Modeling)

- samo modeliranje in dokumentiranje, kombinacija z drugimi
- skupek splošnih principov, vrednost in izkušenj učinkovitega, enostavnega in izboljšujočega modeliranja
- iščemo dobro in ne optimalno rešitev – optimalni so pristopi in orodja

Agilne metodologije: Pregled

Pregled ostalih metodologij in njihovih glavnih idej (nadaljevanje):

c) Prilagodljiv razvoj PO – ASD (ang. Adaptive Software Development)

- bistvo je stalno upoštevanje sprememb
- cikel predvidevanje-sodelovanje-učenje
- ključna komunikacija in ciljno usmerjen, postopen razvoj
- funkcioni razvoj v časovnem okviru z upošte vanjem tveganja in sprememb

d) Družina metodologij Crystal

- veliko metodologij in principov za izbiro
- prilagodljivost – vedno izberemo najprimernejšo(e)
- 3 kriteriji: težavnost uporabe (barve), tveganost ob izgubi podatkov (CDEL), zahtevnost glede osebja (številka)
- Osredotočenost na človeka, minimizacija birokracije, prilagajanje

Agilne metodologije: Pregled

Pregled ostalih metodologij in njihovih glavnih idej (nadaljevanje):

e) Funkcijsko voden razvoj PO – FDD (ang. Feature Driven Development)

- najboljša praksa iz industrije, bolj za modeliranje in razvoj
- več zaporednih procesov: sistem, razbitje na funkcije, razvoj funkcij
- kvaliteten, hiter razvoj po funkcijah

f) Metoda dinamičnega razvoja sistemov – DSDM (ang. Dynamic Systems Development Method)

- obratna filozofija – namesto fiksne funkcionalnosti in spremenljivih virov razvijamo optimalno funkcionalnost z fiksnimi viri (časom, sredstvi)
- 7 faz, veliko študija izvedljivosti
- vključitev uporabnikov, pooblastila skupini, kriterij uspešnosti je izdelek
- celovitost razvoja (zahteva-izdelek), testiranje

Agilne metodologije: Pregled

Pregled ostalih metodologij in njihovih glavnih idej (nadaljevanje):

g) "Vitek" razvoj PO – LSD (ang. Lean Software Development)

- navodila za optimiziranje razvoja avtomobilske industrije (Toyota)
- principi: brez odpadkov, min skladiščenje, max pretok, učenje, JIT, pogosto-pravočasno izročanje, čimkasnejše-pravočasno odločanje same skupine, zagotavljanje kakovosti, pregled celote ...

h) Testno voden razvoj – TDD (ang. Test-Driven Design)

- Stil, kjer testiranje podpira in določa razvoj PO
- najprej testni razredi in nato koda

Uporaba metodologij

- Bolj kakovosten končni izdelek
  - boljši zajem zahtev naročnika
  - boljša dokumentacija
  - boljša združljivost z drugimi sistemi
- Bolj kakovosten proces razvoja
  - stalen vpogled v razvojni proces
  - povezava z modeli za zagotavljanje kakovosti, npr. CMMI, ISO 9001
- Standardizacija poslovanja
  - skupna baza znanja
  - enostavnejše razporejanje ljudi po projektih
- Lažje delo s kadri, izobraževanje kadrov
  - enostavnejše vključevanje novincev
- Vtis na naročnika

Razmerje med agilnostjo in disciplino

- **Klasifikacija metodologij**
  - zgodovinski razvoj:
    - strukturne metodologije -
    - objektno usmerjene metodologije
    - agilne metodologije
  - obseg in gostota:
    - težke metodologije
    - lahke metodologije
  - B. Boehm:
    - tradicionalne metodologije: razvoj po vnaprej pripravljenem planu in vnaprej definiranem postopku
    - agilne metodologije: bistvo prilagodljivost
    - uspešni projekti potrebujejo oboje: agilnost in disciplino

Razmerje med agilnostjo in disciplino

- **Klasifikacija metodologij - zgodovinski razvoj:**
  - strukturne metodologije
  - objektno usmerjene metodologije
  - agilne metodologije
- **Klasifikacija metodologij - obseg in gostota:**
  - težke metodologije
  - lahke metodologije
- **Klasifikacija metodologij - B. Boehm (2003):**
  - tradicionalne metodologije: razvoj po vnaprej pripravljenem planu in vnaprej definiranem postopku
  - agilne metodologije: bistvo prilagodljivost
  - uspešni projekti potrebujejo oboje: agilnost in disciplino
  - zagovorniki obeh pristopov vnašajo zmedo, a tudi vprašanja:
    - disciplina (dokumentacija, prepočasen odziv na spremembe, ...)
    - agilnost (premalo dokumentacije, formalna potrditev razvoja, ...)

Razmerje med agilnostjo in disciplino

Hibridna rešitev:

- vzroki za zmedo
- analiza razlik med obema pristopoma
- možnosti za oblikovanje hibridnih rešitev

Vzroki za zmedo

- različne definicije posameznih pojmov (disciplina: skladnost s procesom ali samokontrola, agilnost: nestanovitnost ali spretnost, kakovost: skladnost s specifikacijo ali zadovoljstvo naročnika)
- zloraba posameznih metod pretirano posploševanje na podlagi posameznih primerov
- zahteve po univerzalnosti ("one size fits all" je mit)
- zgodbe o zgodnjem uspehu
- a je treba metodologijo uporabiti točno tako, kot je predpisana?

Razmerje med agilnostjo in disciplino

Analiza razlik po področjih (agilnost : disciplina):

- Značilnosti aplikacije:

– najpomembnejši cilji:
hitri rezultati in odzivnost : predvidljivost, stabilnost, jamstvo
– velikost projekta (število razvijalcev):
majhni projekti : veliki projekti
– razvojno okolje:
turbulentno, zelo spremenljivo, projektno usmerjeno : mirno, stabilno, organizacijsko usmerjeno

Razmerje med agilnostjo in disciplino

Analiza razlik po področjih (agilnost : disciplina):

Način vodenja:

- razmerje z naročnikom:
  stalno prisoten, najprej doseči prioritetne cilje : po potrebi, pogodbeno določeni cilji
- planiranje in nadzor:
  interni plani, kvalitativni nadzor : dokumentirani plani, kvantitativni nadzor
- komunikacija znotraj projekta:
  izkustvena znanja : dokumentirano znanje

Razmerje med agilnostjo in disciplino

Analiza razlik po področjih (agilnost : disciplina):

- Tehnične značilnosti:

- specifikacija zahtev :
  neformalne zugodbe in testni primeri, ovrednosteni po prioriteti, upoštevane nepredvidljive spremembe : formalna specifikacije, ugotovljena možnost povezave z drugimi sistemi, upoštevane le znane zahteve
- razvoj:
  enostaven načrt, kratki cikli, preoblikovanje kode je poceni : obsežen načrt, daljši cikli, spremebe kode so drage
- testiranje:
  vnaprej pripravljeni avtomatski testi : dokumentiran plan in postopki testiranja

Razmerje med agilnostjo in disciplino

Analiza razlik po področjih (agilnost : disciplina):

- Značilnosti osebja:
  - naročniki: stalno prisoten CRACK naročnik : po potrebi prisoten CRACK naročnik
  - razvijalci: 30% polno zaposlenih 2-3 nivoju, nobenen na 1B ali -1 po Cockburnu : na začetku 50%, ves čas 10% na 3 nivoju, lahko 30% na 1B in nobeden na -1 nivoju po Cockburnu
  - kultura organizacije: kaos – udobje in motivacija temelji na svobodi : red – udobje in motivacijo zagotavlja ustrezna politika in postopki

Razmerje med agilnostjo in disciplino

Nivoji po Cockburnu z opisi

3	Sposoben spremeniti metodo in jo prilagoditi neznani novi situaciji
2	Sposoben prilagoditi metodo znani novi situaciji
1A	S primernim usposabljanjem sposoben izvajati posamezne metodološke korake, npr. izvajati zahtevnejše prestrukturiranje kode. Z izkušnjami lahko preide na nivo 2.
1B	S primernim usposabljanjem sposoben izvajati posamezne postopke metode, npr. kodiranje enostavnih metod ali izvajanje testov. Z izkušnjami lahko pridobi sposobnosti nivoja 1A.
-1	Ima lahko tehnične sposobnosti, vendar ni sposoben ali noče sodelovati ali slediti predpisanim metodam.

Razmerje med agilnostjo in disciplino

Pet kritičnih faktorjev za izbor vrste metodologije (agilnost : disciplina):

- Velikost projekta: prilagojene majhnim projektom in razvojnim skupinam, izkustveno znanje omejuje skalabilnost : prilagojene velikim projektom in razvojnih skupinam, težko prilagodljive za majhne projekte
- Kritičnost: niso preizkušene na projektih z veliko stopnjo varnosti, malo dokumentacije in enostavni načrti so lahko vir težav : za izdelavo varnostno zahtevnih in zanesljivih projektov, težko prilagoditi za nezahtevne projekte
- Dinamičnost: učinkovite v dinamičnih, spremenljivih okoljih, v stabilnih okoljih lahko vzrok za drage popravke : učinkovite v stabilnih okoljih, spremenljiva okolja lahko zahtevajo drage popravke
- Osebje: stalno prisotna kritična masa razvijalcev na 2-3 nivoju, uvajanje razvijalcev na 1B nivoju je tvegano : kritična masa ekspertov potrebna le na začetku, pozneje lahko sodelujejo razvijalci na 1B nivoju po Cockburnu
- Kultura organizacije: udobje in motivacija temelji na svobodi : udobje in motivacijo zagotavlja ustrezna politika in postopki

Razmerje med agilnostjo in disciplino

Grafičen prikaz z polarnim grafom s petimi osmi:

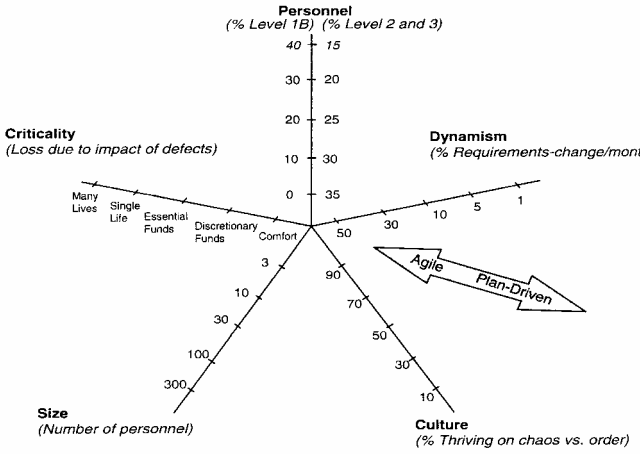


Figure 2-2 Dimensions Affecting Method Selection

Razmerje med agilnostjo in disciplino

Izbira vrste metode s pomočjo grafa:

- Dinamičnost in Osebje sta asimetrični: pri prvem je agilnost uspešna na obeh koncih, disciplina le na enem, pri drugem pa disciplina na obeh koncih, agilnost samo na enem
- vse ocene blizu središča : agilni pristop
- vse ocene na obrobju : discipliniran pristop
- vsaj ena ocena odstopa od ostalih: potrebna je ocena tveganja in uporaba mešanice agilnega in planskega pristopa

Oblikovanje hibridnega pristopa:

- zahteven proces, kiupošteva tehnologijo, vodenje projektov in osebje
- zahteva usposobljene strokovnjake, poznavanje metodologij, okolja in organizacijskih sposobnosti
  - kritični dejavnih uspeha je sposobnost identifikacije in sodelovanja z vsemi zainteresiranimi

Agilne metodologije: Literatura

1. Martin, Robert C.: »Agile Processes«, Agile Development: Principles, Patterns and Process, Prentice Hall, oktober 2002.
2. Beck, A.; Cockburn, A.; Jeffries, R.; Highsmith, J.,: »Agile Manifesto«, http://www.agilemanifesto.org, 2001.
3. Basili, V.; Turner, A. J.: »Iterative Enhancement: A Practical Technique for Software Development«, IEEE Transactions on Software Engineering, vol. 1, no. 4, pp. 390 – 396, 1975.
4. Cohen, D.; Lindvall, M.; Costa, P.: »Agile Software Development – A DACS State–of–the–Art Report«, 2003.
5. Abrahamsson, P.; Salo, O.; Ronkainen, J.; »Warsta, J.: Agile Software Development Methods, Review and Analysis«, VTT 2002.
6. Highsmith, J.; Cockburn, A.: »Agile Software Development: The Business of Innovation«, Computer 34(9), pp. 120-122.

Agilne metodologije: Literatura

7. Ambler, S. W.: »Agile Data Home Page«, http://www.agiledata.org, 2002-2005.
8. Schwaber, K.; Beedle, M.: »Agile Software Development With Scrum«, Upper Saddle River, NJ, Prentice-Hall.
9. Schwaber, K.: »Agile Project Management with Scrum«, Microsoft Press, 2004.
10. Ambler, S. W.: »Agile Modeling (AM) Home Page: Effective Practices for Modeling and Documentation«, http://www.agilemodeling.com.
11. Palmer, S. R.; Felsing, J. M.: »A Practical Guide to Feature-Driven Development«. Upper Saddle River, NJ, Prentice-Hall.
12. Beck, K.: »Extreme Programming Explained: Embrace the Change«, Reading, Mass., Addison-Wesley.
13. B. Boehm, R. Turner: Balancing Agility and Discipline, A Guide for the Perplexed, Addison-Wesley, 2003
14. Internet ...