Prevod knjige „Clean Agile”

Čisto agilno

Razvoj softvera

Agilne vrednosti i principi

kompjuter biblioteka

Robert C. Martin

Čisto agilno

Razvoj softvera

Agilne vrednosti i principi

Za svakog programera koji se ikada borio sa vetrenjačama

Robert C. Martin

kompjuter
biblioteka

Izdavač:

kompjuter biblioteka

Obalskih radnika 4a, Beograd

Tel: 011/2520272

e-mail: kombib@gmail.com

internet: www.kombib.rs

Urednik: Mihailo J. Šolajić

Za izdavača, direktor:
Mihailo J. Šolajić

Autor: Robert C. Martin

Prevod: Biljana Tešić

Lektura: Miloš Jevtović

Slog: Zvonko Aleksić

Znak Kompjuter biblioteke:
Miloš Milosavljević

Štampa: „Pekograf“, Zemun

Tiraž: 500

Godina izdanja: 2021.

Broj knjige: 541

Izdanje: Prvo

ISBN: 978-86-7310-564-2

Clean Agile

Back to Basics

Robert C. Martin

ISBN: 978-0-13-578186-9

Pohvala za knjigu „Čisto agilno“

“Ujak Bob“ je bio na putu ka svim agilnim „stvarima“ i ima „majicu i ožiljke“ da vam to dokaže. Ova divna knjiga delimično je istorija, delimično lična priča i sva mudrost. Ako želite da razumete šta je agilnost i kako je nastala, ova knjiga je za vas.

Grady Booch

Bobova frustracija „boji“ svaku rečenicu u knjizi „Čista agilnost“, ali to je opravdana frustracija. Ono što *postoji* u svetu agilnog razvoja nije ništa u poređenju sa onim što *bi moglo da bude*. Ova knjiga je Bobova perspektiva na šta treba da se fokusirate da biste došli do „onoga što bi moglo da bude“. A Bob je bio na agilnom „putovanju“, pa ga vredi saslušati.

Kent Beck

Dobro je pročitati stav „ujka Boba“ o agilnosti. Bez obzira da li ste tek na početku ili ste sezonski *agilista*, bilo bi dobro da pročitate ovu knjigu. Slažem se sa skoro svim onim što je Bob napisao u knjizi. To su samo neki delovi zbog koji sam shvatio neke svoje nedostatke. Zbog toga sam još jednom proverio pokrivenost kodom (85,09 odsto).

Jon Kern

Ova knjiga obezbeđuje istorijski „objektiv“ kroz koji se može potpunije i tačnije sagledati agilni razvoj. „Ujak Bob“ je jedan od najpametnijih ljudi koje poznajem i bezgranično je oduševljen programiranjem. Ako neko može demistifikovati agilni razvoj, to može on.

Iz „Uvodne reči“, Jerry Fitzpatrick

Kratak sadržaj

Sadržaj

Uvodna reč

Šta je, zapravo, agilni razvoj? Kako je nastao? Kako je evoluirao?

U ovoj knjizi „ujak Bob“ daje promišljene odgovore na ta pitanja. Takođe identifikuje mnoge načine na koje je agilni razvoj pogrešno protumačen ili poremećen. Njegova perspektiva je relevantna, jer je autoritet u toj tematici i jer je učestvovao u „rađanju“ agilnog razvoja.

Bob i ja smo prijatelji dugi niz godina. Upoznali smo se kada sam se pridružio telekomunikacionom sektoru kompanije „Teradyne“ 1979. godine. Kao elektroinženjer, pomagao sam u instaliranju i podršci proizvoda; kasnije sam postao dizajner hardvera.

Otprilike godinu dana nakon što sam se pridružio tom sektoru, kompanija „Teradyne“ je počela da traži ideje o novim proizvodima. Bob i ja smo 1980. godine predložili proizvod „electronic telephone receptionist“, koji je, u osnovi, sistem govorne pošte sa funkcijama usmeravanja poziva. Kompaniji se svideo koncept i ubrzo smo počeli da razvijamo sistem „E.R – The Electronic Receptionist“. Naš prototip je bio najsavremeniji. Pokretao je MP/M operativni sistem na procesoru Intel 8086. Glasovne poruke su smeštene na čvrstom disku Seagate ST-506 od pet megabajta. Dizajnirao sam hardver za glasovni port, a Bob je počeo da piše aplikaciju. Kada sam završio svoj dizajn, napisao sam i kod aplikacije i od onda sam programer.

Otprilike 1985. ili 1986. godine kompanija „Teradyne“ je naglo zaustavila razvoj E.R.-a i zbog nama nepoznatog razloga povukla prijavu patenta. Bila je to poslovna odluka zbog koje je ova kompanija uskoro zažalila i koja još uvek „proganja“ Boba i mene.

Na kraju smo obojica napustili „Teradyne“ zbog drugih poslovnih prilika. Bob je započeo konsultantski posao u okolini Čikaga, a ja sam postao softverski izvođač i instruktor. Uspeli smo da ostanemo u kontaktu, iako sam se preselio u drugu državu.

Do 2000. godine predavao sam objektno-orijentisanu analizu i dizajn u kompaniji „Learning Tree International“. Kurs je obuhvatao UML i Unified Software Development Process (USDP). Bio sam dobro upućen u ove tehnologije, ali ne i u Scrum, ekstremno programiranje, ili slične metodologije.

U februaru 2001. godine objavljen je „Agile Manifest“. Baš kao i među mnogim programerima, moja početna reakcija je bila „agilno šta?“ Jedini manifest za koji sam znao bio je „Manifest komunističke partije“ Karla Marksa, strastvenog komuniste. Da li je ova agilna „stvar“ bila poziv na oružje? Prokleti softverski radikali!

„Agile Manifest“ je pokrenuo svojevrsnu pobunu. Trebalo je da inspiriše razvoj urednog, čistog koda korišćenjem kolaborativnog, prilagodljivog pristupa koji je vođen povratnim informacijama. „Agile Manifest“ je nudio alternativu „teškim“ procesima, kao što su Waterfall i USDP.

Prošlo je 18 godina od objavljivanja „Agile Manifesta“. To je, dakle, drevna istorija većine današnjih programera. Zbog toga, vaše razumevanje agilnog razvoja možda se neće poklapati sa namerom njegovih stvaralaca.

Ova knjiga treba da razjasni agilni razvoj. Obezbeđuje istorijski „objektiv“ kroz koji se može potpunije i tačnije sagledati agilni razvoj. „Ujak Bob“ je jedan od najpametnijih ljudi koje poznajem i bezgranično je oduševljen programiranjem. Ako neko može demistifikovati Agile razvoj, to može on.

Jerry Fitzpatrick

Software Renovation Corporation

mart, 2019.

Predgovor

Ova knjiga nije istraživačko delo. Nisam obavio marljiv pregled literature. Ovo što ćete pročitati su moja lična sećanja, zapažanja i mišljenja o mom dvadesetogodišnjem korišćenju agilnog razvoja softvera - ništa više, ništa manje.

Stil pisanja je neformalan i kolokvijalan. Moj izbor reči je ponekad pomalo grub. I mada nisam neko ko se zaklinje, jedna (malo izmenjena) psovka „ušla" je na ove stranice, jer nisam mogao da smislim bolji način da prenesem pravo značenje.

O ovoj knjizi se neće oduševljeno pričati. Kada sam ih smatrao neophodnim, naveo sam neke reference koje bi trebalo da sledite. Uporedio sam neke od svojih činjenica sa onim drugim činjenicama koje se nalaze u Agile zajednici koliko se u njoj nalazim i ja. Čak sam tražio od nekoliko ljudi da obezbede dodatne i protivurečne tačke gledišta u mojim poglavljima i odeljcima. Ipak, ne bi trebalo da smatrate ovu knjigu naučnim radom. Možda bi bilo bolje da o njoj razmišljate kao o memoarima, tj. kao o gunđanju namćora koji poručuje svoj agilnoj deci da odu sa njegovog travnjaka.

Ova knjiga je namenjena programerima i onima koji nisu programeri. Nije tehnička. Nema koda. Sadrži pregled prvobitne namere agilnog razvoja softvera, bez upuštanja u bilo kakve duboke tehničke detalje programiranja, testiranja i upravljanja.

Ovo je mala knjiga. To je zato što tema nije baš velika. Agilnost je mala ideja o malom problemu malih programerskih timova koji obavljaju male poslove. Agilnost *nije* velika ideja o velikom problemu velikih programerskih timova koji vrše velike poslove. Ironično je što ovo malo rešenje malog problema ima naziv. Na kraju, mali problem o kome je reč rešen je pedesetih i šezdesetih godina prošlog veka, skoro odmah nakon što je osmišljen softver. Onda su mali softverski timovi prilično dobro naučili da obavljaju male poslove. Međutim, sve je „iskočilo iz šina" sedamdesetih godina prošlog veka kada su se mali softverski timovi koji obavljaju male poslove „zapetljali" u ideologiji prema kojoj je trebalo da veliki timovi obavljaju velike poslove.

Zar ne bi trebalo da velike poslove obavljaju veliki timovi? Zaboga, ne! Velike ekipe ne obavljaju velike poslove; veliki poslovi se vrše u saradnji sa mnogih malih timova koji se bave malim poslovima. To su programeri pedesetih i šezdesetih godina prošlog veka instinktivno znali. Međutim, to je zaboravljeno sedamdesetih godina prošlog veka.

Zašto je zaboravljena ideja o malim timovima? Pretpostavljam da se to desilo zbog diskontinuiteta. Broj programera u svetu počeo je da raste pedesetih godina prošlog veka. Pre toga, na svetu je bilo samo nekoliko hiljada programera. Posle toga, bilo ih je stotine hiljada. Sada ih ima blizu sto miliona.

Prvi programeri još pedesetih i šezdesetih godina prošlog veka nisu bili mladi. Programiranjem su počeli u svojim 30-im, 40-im i 50-im godinama. Do sedamdesetih godina prošlog veka taman kad je populacija programera počela da raste, ti stariji programeri su počeli da se penzionišu. Dakle, potrebna obuka se nikada nije dogodila. Veoma mlada kohorta dvadesetogodišnjaka postala je radna snaga baš onda kada su iskusni ljudi odlazili, a njihovo iskustvo nije efektivno preneto na mlade.

Neki ljudi bi rekli da je ovaj događaj označio početak svojevrsno mračnog doba u programiranju. Trideset godina smo se borili sa idejom da velike poslove treba da obavljaju veliki timovi, ne znajući da je tajna u tome da se mnogi mali poslovi obavljaju u mnogo malih timova.

Onda smo sredinom devedesetih godina prošlog veka počeli da shvatamo šta smo izgubili. Ideja o malim timovima počela je da „klija i raste“. Širila se zajednicom softverskih programera, hvatajući tempo. Shvatili smo da nam je potrebno ponovno pokretanje te ideje 2000. godine. Trebalo nas je podsetiti na ono što su naši preci instinktivno znali. Morali smo da shvatimo da velike poslove obavljaju mnogi mali timovi koji međusobno sarađuju i obavljaju male poslove.

Da bismo ovo lakše popularizovali, dali smo ideji naziv. Nazvali smo je „agilna“.

Ovaj predgovor sam napisao prvih dana 2019. godine. Prošle su dve decenije od ponovnog pokretanja agilne ideje 2000. godine i čini mi se da je vreme za još jedno ponovno pokretanje. Zašto? Zato što je jednostavna i mala poruka o agilnosti postala zbrkana tokom ovih godina. Mešana je sa konceptima Lean, Kanban, LeSS, SAFe, Modern, Skilled i mnogim drugim. Ove druge ideje nisu loše, ali nisu ni originalna agilna poruka.

Dakle, vreme je da se još jednom podsetimo na ono što su naši preci znali tokom pedesetih i šezdesetih godina prošlog veka i na ono što smo naučili 2000. godine. Vreme je da se podsetimo šta je agilnost.

U ovoj knjizi nećete naći ništa posebno novo, ništa zapanjujuće ili iznenađujuće, niti bilo šta revolucionarno što razbija kliše. Ono što ćete naći je prilagođavanje agilne ideje iz 2000. godine, koja je osmišljena iz druge perspektive, a mi smo tokom poslednjih 20 godina naučili nekoliko „stvari" koje ću uključiti u knjizi. U celini, poruka ove knjige je poruka iz 2001. i 1950. godine.

To je stara poruka. To je istinita poruka. To je poruka koja nam obezbeđuje malo rešenje za mali problem malih softverskih timova koji obavljaju male poslove.

Registrujte svoj primerak knjige „Čisto agilno" na sajtu InformIT zbog pogodnosti pristupa ažuriranjima i/ili ispravkama čim postanu dostupne. Da biste započeli postupak registracije, posetite sajt informit.com/register i prijavite se ili kreirajte nalog. Unesite ISBN proizvoda (9780135781869) i kliknite na Submit. Na kartici Registered Products potražite link Access Bonus Content pored naslova ove knjige i sledite taj link da biste pristupili svim dostupnim bonus materijalima. Ako želite da budete obavešteni o ekskluzivnim ponudama za nova izdanja i ažuriranja, potvrdite izbor da bismo vam slali e-poštu.

Zahvalnice

Prvo se zahvaljujem dvojici neustrašivih programera koji su radosno otkrili (ili ponovo otkrili) tehnike koje su ovde sadržane: Vardu Cunninghamu i Kentu Becku.

Sledeći na listi ljudi kojima dugujem zahvalnost je Martin Fowler, bez čije bi mirne ruke, u tim najranijim danima, agilna revolucija verovatno bila mrtvorođena.

Zahvaljujem se Kenu Schwaberu zbog nesalomive energije koju je primenio u promociji i usvajanju agilnog razvoja.

Mary Poppendieck takođe zaslužuje posebno priznanje zbog nesebične i neiscrpne energije koju je uložila u Agile pokret i u upravljanje savezom Agile Alliance.

Po mom mišljenju, Ron Jeffries je, kroz svoje razgovore, članke, blogove i postojane odlike svog karaktera, bio glas razuma ranog Agile pokreta.

Mike Beedle vodio je dobru borbu za agilnost, ali neki beskućnik ga je besmisleno ubio na ulicama Čikaga.

Ostali originalni autori „Agile Manifesta“ zauzimaju ovde posebno mesto:

Arie van Bennekum, Alistair Cockburn, James Grenning, Jim Highsmith, Andrew Hunt, Jon Kern, Brian Marick, Steve Mellor, JeffSutherland i Dave Thomas.

JimNewkirk, moj prijatelj i poslovni partner u to vreme, neumorno je radio da bi podržao Agile dok se suočavao sa problemima koje većina nas ne možemo da zamislimo.

Takođe je potrebno da pomenem ljude koji su radili u kompaniji „Object Mentor Inc“. Svi oni su preuzeli na sebe rizik da usvoje i promovišu agilnost. Mnogi od njih su na sledećem snimku koja je napravljen na početku prvog kursa XP Immersion.

drugi red: Ron Jeffries, autor, Brian Button, Lowell Lindstrom, Kent Beck, Micah Martin, Angelique Martin, Susan Rosso i James Grenning

prvi red: David Farber, Eric Meade, Mike Hill, Chris Biegay, Alan Francis, Jennifer Kohnke, Talisha Jefferson i Pascal Roy

Nisu na slici: Tim Ottinger, Jeff Langr, Bob Koss, Jim Newkirk, Michael Feathers, Dean Wampler i David Chelimsky

Takođe se zahvaljujem ljudima koji su se okupili da formiraju savez Agile Alliance. Neki od njih su na sledećem snimku koji je napravljen na prvom važnom sastanku saveta.

sleva nadesno: Mary Poppendieck, Ken Schwaber, autor, Mike Beedle,
Jim Highsmith (nije na slici Ron Crocker)

Na kraju, hvala svim ljudima iz Pearsona, posebno mom izdavaču Julie Phifer!

O autoru

Robert C. Martin „ujak Bob" je programer od 1970. godine. On je suosnivač sajta cleancoders.com, koji obezbeđuje video obuku za softverske programere na Internetu, i osnivač je firme „Uncle Bob Consulting LLC", koja obezbeđuje softversko savetovanje i obuku i podučavanje u razvojnoj veštini u velikim korporacijama širom sveta. Ranije je služio kao „majstor svog zanata" u konsultantskoj softverskoj firmi „8th Light Inc.", čije je sedište u Čikagu.

Martin je objavio desetine članaka u raznim trgovačkim časopisima i redovan je govornik na međunarodnim konferencijama i prodajnim sajmovima. Napisao je i uredio mnoge knjige, uključujući:

Designing Object-Oriented C++ Applications Using the Booch Method

Patterns Languages of Program Design 3

More C++ Gems

Extreme Programming in Practice

Agile Software Development: Principles, Patterns, and Practices

UML for Java Programmers

Clean Code

The Clean Coder

Clean Architecture

Clean Agile

Robert C. Martin, lider u industriji razvoja softvera, tri godine je bio glavni urednik časopisa „C++ Report“ i prvi predsedavajući saveza „Agile Alliance“.

Postanite član Kompjuter biblioteke

Kupovinom jedne naše knjige stekli ste pravo da postanete član Kompjuter biblioteke. Kao član možete da kupujete knjige u pretplati sa 40% popustai učestvujete u akcijama kada ostvarujete popuste na sva naša izdanja. Potrebno je samo da se prijavite preko formulara na našem sajtu.
Link za prijavu: http://bit.ly/2TxekSa

Skenirajte QR kod
registrujte knjigu
i osvojite nagradu

1 Uvod u agilnost

U februaru 2001. godine grupa od 17 softverskih stručnjaka okupila se u Snowbirdu, u državi Juti, da bi razgovarala o jadnom stanju razvoja softvera. U to vreme većina softvera je kreirana korišćenjem neefikasnih, teških, visokoritualnih procesa, kao što su Waterfall, i prenapunjenih instanci Rational Unified Processa (RUP-a). Cilj ovih 17 stručnjaka bio je da se kreira manifest koji će uvesti efikasniji i laganiji pristup.

To nije bio nikakav podvig. Ovih 17 stručnjaka bili su ljudi različitog iskustva i snažnih divergentnih mišljenja. Za mogućnost da će takva grupa doći do konsenzusa postojala je mala verovatnoća. Ipak, uprkos lošim šansama, postignut je konsenzus, tj. napisan je „Agile Manifest" i „rođen" je jedan od najmoćnijih i najdugovečnijih pokreta u oblasti razvoja softvera.

Pokreti u oblasti razvoja softvera prate predvidiv put. U početku postoji jedna manjina entuzijastičnih pristalica, druga manjina entuzijastičnih klevetnika i velika većina onih koji su ravnodušni. Mnogi pokreti umiru u toj fazi ili je bar nikad ne napuštaju. Zamislite aspektno-orijentisano programiranje, ili logičko programiranje ili CRC kartice. Neki pokreti, međutim, pređu „provaliju" i postanu izuzetno popularni i kontroverzni. Neki čak uspevaju da ostave kontroverzu iza sebe i jednostavno postanu deo mejnstrima. Objektna-orijentacija (OO) je postala deo mejnstrima, ali i agilnost.

Nažalost, kada neki pokret postane popularan, naziv pokreta se „zamagljuje" nerazumevanjem i uzurpacijom. Autori proizvoda i metoda koji nemaju nikakve veze sa stvarnim pokretom pozajmljivaće naziv da bi unovčili popularnost i značaj u ime tog pokreta. Tako je bilo i sa Agile pokretom.

Ovde su predstavljene osnove agilnosti. Mnogi su ulepšali i proširili ove ideje agilnosti i u tome nema ničeg lošeg. Međutim, ta proširenja i ukrasi nisu samo agilni. Ovde ćete saznati šta je agilnost sada, šta je bila i šta će neizbežno uvek da bude.

Istorija agilnosti

Kada je agilnost započeta? Verovatno pre više od 50.000 godina, kada su ljudi prvi put odlučili da sarađuju na zajedničkom cilju. Ideja izbora malih srednjih ciljeva i merenja napretka nakon ostvarenja svakog od njih je previše intuitivna i previše ljudska da bi se mogla smatrati bilo kakvom revolucijom.

Kada je agilnost primenjena u modernoj industriji? Teško je reći. Pretpostavljam da su prva parna mašina, prvi mlin, prvi motor sa unutrašnjim sagorevanjem i prvi avion proizvedeni pomoću tehnika koje bismo sada nazvali agilnim. Razlog tome je što je preduzimanje malih odmerenih koraka previše prirodno i ljudski da bi se to moglo dogoditi na bilo koji drugi način.

Dakle, kada je agilnost primenjena u razvoju softvera? Voleo bih da sam mogao da budem muva na zidu dok je Alan Turing pisao svoj rad 1936. godine.[1] Pretpostavljam da su mnogi „programi" koje je napisao u svojoj knjizi razvijeni malim koracima, na onovu mnogo pregleda dokumentacije. Takođe pretpostavljam da je prvi kod koji je Turing napisao za kompaniju „Automatic Computing Engine" 1946. godine napisan pomoću malih koraka, pomoću mnogo pregleda dokumentacije, pa čak i pomoću nekog stvarnog testiranja.

Rani dani razvoja softvera puni su primera ponašanja koje bismo sada opisali kao agilno. Na primer, programeri koji su napisali kontrolni softver za svemirsku kapsulu „Merkur" radili su pomoću poludnevnih koraka koji su bili isprekidani jediničnim testovima.

Mnogo je o ovom periodu pisano negde drugde. Craig Larman i Vic Basili napisali su istoriju koja je rezimirana na wikiju Warda Cunninghama[2], a takođe i u Larmanovoj knjizi „Agile & Iterative Development: A Manager's Guide".[3]

---

1 Turing, A. M. 1936. O izračunljivim brojevima, sa primenom procedure Entscheidungsproblem [dokaz]. „Proceedings of the London Mathematical Society, 2" (objavljeno 1937), 42(1):230–65. Najbolji način za razumevanje ovog rada je čitanje remek-dela Charlesa Petzolda: Petzold, C. 2008. „The Annotated Turing: A Guided Tour through Alan Turing's Historic Paper on Computability and the Turing Machine". Indianapolis, IN: Wiley

2 Wardov wiki, c2.com, je originalni wiki - prvi koji se pojavio na Internetu. Neka nam dugo služi.

3 Larman, C. 2004. „Agile & Iterative Development: A Manager's Guide". Boston, MA: Addison-Wesley.

Međutim, agilnost nije bila jedina „igra u gradu". Zaista, postojala je konkurentska metodologija koja je uživala znatan uspeh u proizvodnji i industriji uopšte: to je naučni menadžment (Scientific Management).

Naučni menadžment je pristup odozgo prema dole, komandovanje i upravljanje. Menadžeri koriste naučne tehnike da bi utvrdili najbolje procedure za postizanje cilja, a zatim ukazuju svim podređenima da treba doslovno da slede njihov plan. Drugim rečima, postoji veliko planiranje unapred koje je praćeno pažljivom detaljnom implementacijom.

Naučni menadžment je, verovatno, star koliko i piramide, Stounhendž ili neko drugo veliko delo drevnih vremena, jer je nemoguće da su takva dela mogla da nastanu bez njega. Ponavljamo: ideja ponavljanja uspešnog procesa previše je intuitivna i ljudska da bi se mogla smatrati nekom vrstom revolucije.

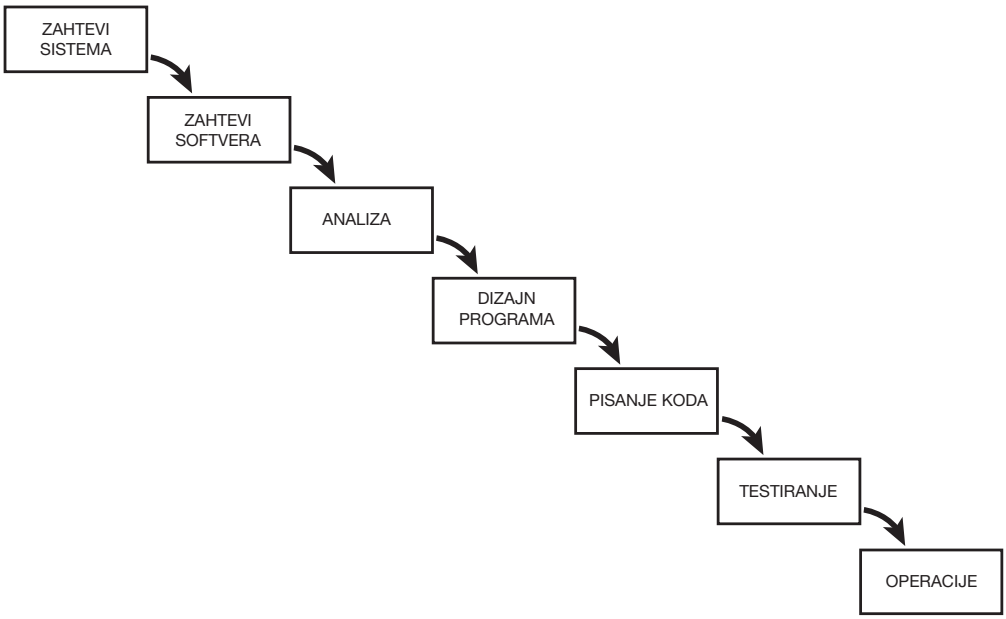
Naučni menadžment je dobio naziv po radovima Fredericka Winslowa Taylora osamdesetih godina 19. veka. Taylor je formalizovao i komercijalizovao pristup i obogatio se kao savetnik menadžmenta. Naučni menadžment je bio izuzetno uspešna tehnika i dovela je do masovnog povećanja efikasnosti i produktivnosti tokom decenija koje su usledile.

Zatim je 1970. godine svet softvera bio na razmeđi ove dve suprotstavljene tehnike. Pre-Agile (Agile pre nego što je nazvan Agile) preduzeo je kratke reaktivne korake koji su bili izmereni i usavršeni da bi se u usmerenom nasumičnom hodu došlo do dobrog ishoda. Naučni menadžment je odložio akciju dok se ne obavi temeljna analiza i kreira detaljan plan. Pre-Agile je bio dobar za projekte koji su uživali u niskim troškovima promena i rešavao je delimično definisane probleme pomoću neformalno navedenih ciljeva. Naučni menadžment je bio najbolji za projekte koji su pretrpeli velike promene i rešavao je veoma dobro definisane probleme pomoću izuzetno specifičnih ciljeva.

Pitanje je bilo koje vrste projekata su bili softverski projekti. Da li su bili projekti sa visokim troškovima promena i dobro definisani, sa određenim ciljevima, ili su bili projekti sa niskim troškovima promena, sa delimično definisanim neformalnim ciljevima?

Nemojte se previše udubljivati u prethodni pasus. Koliko znam, niko zapravo nije postavio to pitanje. Ironično, čini se da je put koji je odabran sedamdesetih godina prošlog veka više vođen slučajno nego namerno.

Winston Royce je 1970. godine napisao rad[4] u kojem je opisao svoje ideje za upravljanje velikim softverskim projektima. Na dijagramu (slika 1.1) je prikazan njegov plan. Royce nije bio začetnik ovog dijagrama, niti ga je preporučio kao plan. Zapravo, dijagram je postavljen kao fiktivna ideja koja je „srušena" na narednim stranicama njegovog rada.



Slika 1.1 Dijagram Winstona Royca koji je bio inspiracija za Waterfall razvoj

Ipak, istaknuto postavljanje dijagrama i sklonost ljudi da zaključuju o sadržaju rada na osnovu dijagrama na prvoj ili drugoj stranici doveli su do dramatičnog pomaka u softverskoj industriji.

4 Roice, W. W. 1970. Upravljanje razvojem velikih softverskih sistema. „Proceedings, IEEE WESCON",avgust: 1–9. Pristupite ovom radu na adresi http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf.

Roycev početni dijagram toliko je ličio na vodu koja se spušta niz nekoliko stena da je tehnika postala poznata kao Waterfall (vodopad).

Waterfall je bio logični potomak naučnog menadžmenta. U njemu se radi detaljna analiza, pravi detaljan plan, a zatim se ispuni taj plan do kraja.

Iako ovaj koncept Royce nije preporučivao, ljudi su ga preuzeli upravo iz Royceovog rada. I dominirao je u naredne tri decenije.[5]

Ovde ja „ulazim u priču". Imao sam 1970. godine 18 godina i radio sam kao programer u kompaniji „A. S. C. Tabela u Lake Bluff" u Ilinoisu. Ta kompanija je imala IBM 360/30 sa 16K jezgrom, IBM 360/40 sa 64K jezgrom i Varian 620/f miniračunar sa 64K jezgrom. Programirao sam 360s na programskim jezicima COBOL, PL/1, Fortran i asembler. Napisao sam samo asembler za 620/f.

U to doba kod smo pisali na obrascima za kodiranje pomoću olovaka i imali smo rukovaoce mašinama za bušenje kartica. Pažljivo proverene kartice predali bismo operaterima računara koji su vršili kompajliranje i testove tokom treće smene, jer su računari tokom dana bili previše zauzeti. Često su bili potrebni dani da bi se stiglo od početnog pisanja koda do prvog kompajliranja, a svaki novi posao nakon toga obično je trajao jedan dan.

Mašina 620/f je za mene bila malo drugačija. Ona je bila namenjena za naš tim, tako da smo joj mogli pristupiti 24 sata dnevno. Mogli smo da izvršimo dva, tri, možda čak i četiri zadatka i ispitivanja dnevno. Tim u kojem sam bio takođe je bio sastavljen od ljudi koji su, za razliku od većine „dnevnih" programera, mogli da kucaju. Dakle, mi smo bušili kartice, umesto da taj posao predamo hirovitim rukovaocima mašinom za bušenje kartica.

5 Treba napomenuti da je moje tumačenje ove vremenske linije osporio Bossavit u Poglavlju 7, L. 2012. „The Leprechauns of Software Engineering: How Folklore Turns into Fact and What to Do About It". Leanpub.

Koji proces smo koristili tih dana? Sigurno nismo koristili Waterfall. Nismo imali koncept da sledimo detaljne planove. Samo smo svakodnevno hakovali, pokretali kompajlere, testirali naš kod i ispravljali greške. Bila je to beskrajna petlja koja nije imala strukturu. Takođe nije postojala agilnost, a čak ni Pre-Agile. Nije bilo discipline u načinu na koji smo radili. Nije bilo niza testova, niti izmerenih vremenskih intervala. To je bio samo pisanje koda i popravljanje, pisanje koda i popravljanje, dan za danom, mesec za mesecom.

O Waterfallu sam prvi put pročitao tekst u trgovinskom časopisu otprilike 1972. godine. Činilo mi se kao da je božji dar. Da li bismo zaista mogli da unapred analiziramo problem, zatim da osmislimo rešenje za taj problem, a onda da primenimo taj dizajn? Da li bismo zaista mogli da kreiramo raspored na osnovu te tri faze? Kada bismo završili analizu, da li bismo zaista završili jednu trećinu projekta? Osetio sam snagu koncepta. Hteo sam da verujem. Jer, ako bi Waterfall bio uspešan, bilo bi to ostvarenje sna.

Očigledno nisam bio sam, jer su i mnogi drugi programeri i prodavnice programa uočili grešku. I, kao što sam već rekao, Waterfall je počeo da dominira načinom na koji smo razmišljali.

Dominirao je, ali nije bio uspešan. Sledećih trideset godina moji saradnici, programeri širom sveta i ja pokušavali smo da ispravimo tu analizu i dizajn. Međutim, svaki put kada smo pomislili da smo rešili problem, prošao nam je „kroz prste" tokom faze implementacije. Svi naši meseci pažljivog planiranja pred sjajnim očima mendžera i klijenata postali su nevažni zbog neizbežnog utrkivanja i veoma kratkih rokova.

Bez obzira na praktično neprekidan tok neuspeha, ustrajali smo u Waterfall majndsetu. Na kraju krajeva, kako Waterfall može propasti? Kako detaljna analiza problema, pažljivo osmišljavanje rešenja, a zatim primena tog dizajna mogu tako spektakularno propasti iznova? Bilo je nezamislivo (inconceivable)[6] da problem leži u toj strategiji. Problem je morao biti u nama. Nešto smo radili pogrešno.

6 Pogledajte film „The Princess Bride" (1978) da biste čuli promenu te reči.

Nivo do kojeg je dominirao Waterfall majndset može se videti u jeziku tog vremena. Kada je Dijkstra osmislio strukturirano programiranje 1968. godine, strukturna analiza[7] i strukturirani dizajn[8] nisu zaostajali. Kada je objektno-orijentisano programiranje (OOP) počelo da postaje popularno 1988. godine, objektno-orijentisana analiza[9] i objektno-orijentisani dizajn[10] (OOD) takođe nisu zaostajali. Ovaj triplet mema, ovaj trijumvirat faza, uhvatio nas je u zamku. Jednostavno, nismo mogli da zamislimo drugačiji način rada. A onda smo odjednom mogli da ga zamislimo.

Počeci agilne reformacije započeli su krajem osamdesetih ili početkom devedesetih godina prošlog veka. Zajednica Smalltalk počela je da pokazuje znake te reformacije osamdesetih godina prošlog veka. Nagoveštaja o tome bilo je u Boochovoj knjizi o OOD-u iz 1991. godine. Više rezolucija pojavilo se u Cockburnovoj knjizi „Crystal Methods" 1991. godine. Zajednica Design Patterns je o tome počela da raspravlja 1994. godine, podstaknuta radom Jamesa Copliena.[11]

Do 1995. godine Beedle[12], Devos, Sharon, Schwaber i Sutherland napisali su svoj poznati rad na Scrumu[13]. I vrata su se otvorila. „Bastion" Waterfalla je probijen i nije bilo povratka nazad.

Ovde još jednom ja „ulazim u priču". Ono što sledi je iz mog sećanja i nisam pokušao da to sećanje uporedim sa sećanjima ostalih koji su učestvovali u agilnoj reformaciji. Stoga bi trebalo da pretpostavite da ovo moje sećanje ima mnogo propusta i da sadrži mnogo štošta što je apokrifno ili bar prilično neprecizno. Međutim, nemojte paničiti, jer sam bar pokušao da ono čega se sećam predstavim na zabavan način.

---

7 DeMarco, T. 1979. „Structured Analysis and System Specification". Upper Saddle River, NJ: Yourdon Press.

8 Page-Jones, M. 1980. „The Practical Guide to Structured Systems Design". Englewood Cliffs, NJ: Yourdon Press.

9 Coad, P., and E. Yourdon. 1990. „Object-Oriented Analysis. Englewood Cliffs", NJ: Yourdon Press.

10 Booch, G. 1991. „Object Oriented Design with Applications. Redwood City, CA": Benjamin-Cummings Publishing Co.

11 Coplien, J. O. 1995. „A generative development-process pattern language. Pattern Languages of Program Design". Reading, MA: Addison-Wesley, str. 183

12 Mikea Beedlea ubio je 23. marta 2018. godine u Čikagu mentalno poremećeni beskućnik koji je ranije hapšen i puštan 99 puta. Trebao je biti institucionalizovan. Mike Beedle je bio moj prijatelj.

13 Beedle, M., M. Devos, Y. Sharon, K. Schwaber, and J. Sutherland. „SCRUM: An extension pattern language for hyperproductive software development". Knjizi pristupite na adresi http://jeffsutherland.org/scrum/scrum_plop.pdf.

Kenta Becka sam upoznao 1994. godine na PLOP-u[14], na kojem je bio predstavljen Coplienov rad. Bio je to neobavezan sastanak i na njemu se nije baš ništa postiglo. Sreo sam Kenta sledećeg februara 1999. godine na OOP konferenciji u Minhenu. Međutim, do tada sam saznao mnogo više o njemu.

U to vreme sam bio savetnik za C++ i OOD, išao sam svuda, pomagao ljudima i dizajnirao i implementirao aplikacije na C++ jeziku, koristeći OOD tehnike. Klijenti su počeli da me ispituju o procesu. Čuli su da se Waterfall ne meša sa OO i želeli su moj savet. Složio[15] sam se da treba mešati OO i Waterfall i dugo sam razmišljao o ovoj ideji. Čak sam i pomislio da bih mogao da napišem svoj OO proces. Srećom, rano sam odustao od toga, jer sam slučajno naišao na tekstove Kenta Becka o ekstremnom programiranju (XP).

Što sam više čitao o XP-u, to sam bio više fasciniran. Ideje su bile revolucionarne (ili sam bar u to vreme tako mislio). Imale su smisla, posebno u OO kontekstu (ponavljam: tako sam onda mislio). Dakle, bio sam nestrpljiv da naučim više.

Na moje iznenađenje, na toj OOP konferenciji u Minhenu održao sam predavanje preko puta Kenta Becka. Pri susretu tokom pauze dogovorili smo se da se nađemo na ručku da bismo razgovarali o XP-u. Na tom ručku smo kreirali osnovu za značajno partnerstvo. Moji razgovori sa njim naveli su me da odem do Kentove kuće u Medfordu, u Oregonu, da bih sarađivao sa njim na dizajniranju kursa o XP-u. Tokom te posete prvi put sam isprobao razvoj Test Driven Development (TDD) i zainteresovao sam se za njega.

U to vreme vodio sam kompaniju „Object Mentor". Udružio sam se sa Kentom da bismo ponudili petodnevni kurs za obuku o XP-u, koji smo nazvali *XP Immersion*. Od kraja 1999. do 11. septembra 2001. godine[16] kurs je bio veliki hit! Obučili smo stotine ljudi.

14 „Pattern Languages of Programming" bila je konferencija održana devedesetih godinaprošlog veka u blizini Univerziteta Ilinois.

15 Ovo je jedna od onih čudnih slučajnosti koje se s vremena na vreme dogode. Ne postoji ništa posebno u vezi sa OO-mzbog čega ne bi mogao da se meša sa Waterfallom, a opet taj mem je tih dana izazvao veliko interesovanje.

16 Značaj tog datuma ne treba zanemariti.

U leto 2000. godine Kent je pozvao kvorum ljudi iz zajednice XP i Patterns na sastanak u blizini svoje kuće. Nazvao ga je „XP Leadership" sastanak. Vozili smo se čamcima i pešačili obalama reke Ruž. I sreli smo se da bismo odlučili šta ćemo da učinimo u vezi sa XP-om.

Jedna od ideja je bila da oformimo XP neprofitnu organizaciju. Ja sam bio za tu ideju, ali mnogi nisu. Očigledno su imali negativno iskustvo sa sličnom grupom osnovanom za realizaciju ideje o projektnim obrascima. Frustrirano sam napustio taj sastanak, ali Martin Fowler me je pratio i predložio da se nađemo kasnije u Čikagu da bismo razgovarali. Složio sam se.

Zatim smo se Martin i ja sreli u jesen 2000. godine u kafiću u blizini kancelarije „ThoughtWorks", u kojoj je on radio. Opisao sam mu svoju ideju da okupim sve konkurentske zagovornike laganog procesa da bismo formirali manifest jedinstva. Martin je dao nekoliko preporuka za listu pozivnica, a mi smo sarađivali na pisanju pozivnica. Pozivnice sam poslao kasnije tog dana. Tema je bila *Light Weight Process Summit*.

Jedan od pozvanih bio je Alistair Cockburn. Pozvao me je da kaže da se upravo spremao da sazove sličan sastanak, ali da mu se naša lista pozivnica više svidela od njegove. Ponudio je da spoji svoju listu sa našom i da obavi pripreme za sastanak ako se dogovorimo da sastanak održimo na skijalištu Snowbird, blizu Solt Lejk Sitija.

Dakle, zakazan je sastanak u Snowbirdu.

Snowbird

Bio sam prilično iznenađen što se toliko ljudi pojavilo na sastanku u Snowbirdu. Mislio sam ko li zaista želi da prisustvuje sastanku koji se zove „The Light Weight Process Summit". Međutim, evo nas u Aspen sali u loži, u Snowbirdu.

Bilo nas je 17. Od onda nas kritikuju da smo 17 sredovečnih belaca, muškaraca. Kritika je korektna u nekoj meri. Međutim, najmanje jedna žena Agneta Jacobson bila je pozvana, ali nije mogla da prisustvuje.

I, uostalom, velika većina starijih programera u svetu u to vreme bili su sredovečni belci - razlozi su priča za druga vremena i drugu knjigu.

Nas 17 predstavljalo je dosta različitih gledišta, uključujući pet različitih laganih procesa. Najveća kohorta bio je XP tim: Kent Beck, James Grenning, Ward Cunningham, Ron Jeffries i ja. Sledio je Scrum tim: Ken Schwaber, Mike Beedle i Jeff Sutherland. Jon Kern predstavljao je razvoj vođen funkcijama, a Arie van Bennekum predstavljao je Dynamic Systems Development Method (DSDM). Na kraju, Alistair Cockburn je predstavio Crystal porodicu procesa.

Ostatak ljudi je bio relativno nezavisan. Andy Hunt i DaveThomas su bili pragmatični programeri. Brian Marick je bio savetnik za testiranje. Jim Highsmith je bio savetnik za upravljanje softverom. Steve Mellor je bio tu da nas upozori da treba da se potrudimo, jer je zastupao filozofiju koja je vođena modelima, a mnogi od nas su bili sumnjičavi kada je reč o toj filozofiji. I na kraju je došao Martin Fowler, koji je, iako je imao bliske veze sa XP timom, bio skeptičan prema bilo kojoj vrsti brendiranog procesa i saosećao je sa svima.

Ne sećam se mnogo detalja tog dvodnevnog sastanka. Drugi koji su bili na tom sastanku imaju drugačije sećanje od mene.[17] Dakle, reći ću vam samo ono čega se ja sećam i sugerišem vam da to shvatite kao skoro dve decenije staro sećanje 65-godišnjaka. Možda ću propustiti nekoliko detalja, ali suština je verovatno tačna.

Dogovoreno je, nekako, da ja započnem sastanak. Zahvalio sam se svima na dolasku i predložio da naša misija bude kreiranje manifesta koji opisuje ono što smo verovali da je zajedničko svim tim laganim procesima i razvoju softvera uopšte. Verujem da je to bio moj jedini doprinos sastanku.

17 Nedavno je objavljena istorija događaja u članku „The Atlantic": Mimbs Nice, C. 2017. Beg od zime koji je preokrenuo svet softvera. „The Atlantic". 8. decembar. Članku se može pristupiti na adresi https://www.theatlantic.com/technologi/archive/2017/12/agile-manifesto-a-history/547715/. Od ovog pisanja knjiga nisam pročitao taj članak, jer ne želim da „zagađujem" sećanje koje ovde pišem.

Radili smo standardno - zapisivali smo probleme na karticama, a zatim smo sortirali kartice na podu u grupacije afiniteta. Ne znam stvarno da li je to nekuda vodilo. Samo se sećam da sam to radio.

Ne sećam se da li se „magija" dogodila prvog ili drugog dana. Možda su grupacije afiniteta identifikovale četiri vrednosti, a to su pojedinci i interakcije, radni softver, saradnja sa klijentima i odgovor na promene. Neko ih je napisao na tabli u prednjem delu sobe, a zatim je imao sjajnu ideju da kaže da se njima daje prednost, ali da one ne zamenjuju komplementarne vrednosti procesa, alatki, dokumentacija, ugovora i planova.

Ovo je bila centralna ideja „Agile Manifesta", a čini se da niko nije tačno zapamtio ko ju je prvi napisao na tabli. Ja mislim da je to bio Ward Cunningham. Međutim, Ward veruje da je to bio Martin Fowler.

Pogledajte sliku na veb stranici agilemanifesto.org. Ward kaže da je ovaj snimak napravio da bi zabeležio taj trenutak. Na slici se jasno vidi da je Martin kod table, a svi mi smo se okupili oko njega.[18] To ide u prilog Wardovoj tvrdnji da je Martin bio taj koji je osmislio ideju.

Sa druge strane, možda je najbolje da nikada zapravo ne saznamo.

Kada se „magija" dogodila, cela grupa se spojila oko nje. Bilo je tu i stihoklepanja i podešavanja i prilagođavanja. Koliko se sećam, Ward je napisao preambulu: „Otkrivamo bolje načine za razvoj softvera, radeći razvoj softvera i pomažući drugima da ga urade".

18 S leva na desno, u polukrugu oko Martina, na toj slici su Dave Thomas, Andy Hunt (ili možda Jon Kern), ja (možete da vidite po plavim farmerkama i Leathermanu na pojasu), Jim Highsmith, još neko, Ron Jeffries i James Grenning. Neko sedi iza Rona, a na podu kraj njegove cipele izgleda da je jedna od kartica koje smo koristili u grupisanju afiniteta.

Ostali su napravili male izmene i dali svoje sugestije, ali bilo je jasno da smo završili. U sobi se „osećao" završetak. Nije bilo neslaganja. Nije bilo argumenata. Čak nije bilo ni stvarnih diskusija o alternativama. Četiri reda teksta su predstavljala tu „magiju":

- **pojedinci i interakcije** u skladu sa procesima i alatkama
- **radni softver** u skladu sa sveobuhvatnom dokumentacijom
- **saradnja sa klijentima** tokom pregovora o ugovoru
- **odgovor na promenu** u skladu sa planom

Naravno, bilo je mnogo detalja koje je trebalo razumeti. Prvo, kako bismo nazvali ovo što smo identifikovali?

Naziv „agilno" nije bio pogodak „iz prve". Bilo je mnogo različitih pretendenata. Slučajno mi se svidela reč „lagano", ali nikom drugom nije. Mislili su da ta reč podrazumeva „nevažno". Drugima se svidela reč „prilagodljiv". Spomenuta je reč „agilno", a jedna osoba je prokomentarisala da je ta reč trenutno popularna u vojsci. Na kraju, iako niko zapravo nije voleo reč „agilno", zaključili smo da je bila najbolja od gomile loših alternativa.

Pošto se drugi dan bližio kraju, Ward se javio kao dobrovoljac da postavi veb stranicu agilemanifesto.org. Verujem da je bila njegova ideja da prisutni učesnici sastanka potpišu taj manifest.

Posle Snowbirda

Naredne dve nedelje nisu bile ni izbliza toliko romantične, niti nalik na događanja kao ona dva dana u Snowbirdu. Uglavnom je dominiralo naporno iscrtavanje dokumenta sa principima koji je Ward na kraju dodao na veb sajt.

Ideja da napišemo ovaj dokument bila je nešto za šta smo se svi složili da je neophodno da bi se objasnile i usmerile četiri vrednosti. Sve u svemu, četiri vrednosti su vrste iskaza sa kojima se svi mogu složiti, a da, pri tom, zapravo, ništa ne menjaju u svom načinu rada.

Principi jasno pokazuju da te četiri vrednosti imaju dalekosežne posledice.

Nemam mnogo snažnih sećanja o ovom periodu, osim da smo dokument koji sadrži principe međusobno slali u više navrata. Bio je to naporan posao, ali mislim da smo svi osećali da je trud opravdan. Nakon toga smo se svi vratili svojim uobičajenim poslovima i aktivnostima. Pretpostavljam da je većina nas mislila da će se „priča" tu završiti.

Niko od nas nije očekivao ogroman talas podrške koja je usledila. Niko od nas nije predvideo koliko su ta dva dana bila važna. Međutim, da mi ne bi natekla glava što sam bio deo toga, neprestano se podsećam da je Alistair bio na ivici da sazove sličan sastanak. I to me tera da se zapitam koliko je drugih takođe bilo na ivici da učini nešto slično. Tako da se zadovoljavam idejom da je vreme sazrelo i pretpostavkom da se nas 17 nije sastalo na toj planini u Juti neka druga grupa bi se sastala negde drugde i došla bi do sličnog zaključka.

Pregled agilnosti

Kako upravljate softverskim projektom? Tokom godina bilo je mnogo pristupa - većina njih je bila prilično loša. Nada i molitva su popularni među onim menadžerima koji veruju da postoje bogovi koji upravljaju sudbinom softverskih projekata. Oni koji nemaju takvu veru često se vraćaju motivacionim tehnikama, kao što su nametanje datuma „bičevima", „lancima", „kipućim uljem" i slikama ljudi koji se penju na stene i galebova koji lete iznad okeana.

Ovi pristupi skoro univerzalno dovode do karakterističnih simptoma lošeg upravljanja softverom, kao što su razvojni timovi koji uvek kasne iako rade prekovremeno i timovi koji kreiraju proizvode očigledno lošeg kvaliteta, koji ni približno ne zadovoljavaju potrebe kupaca.