ODABIR MODELA ŽIVOTNOG CIKLUSA RAZVOJA SOFTVERA: POSLEDICE NA NAČIN UPRAVLJANJA

MATEMATIČKI FAKULTET, UNIVERZITET U BEOGRADU

PROFESOR: PROF. DR. VLADIMIR FILIPOVIĆ

STUDENT: BOŽIDAR RADIVOJEVIĆ

BEOGRAD, 22.09.2016.

UVOD

(Jer od nečega valja uvek početi)

UVOD

- *„Uspeh u softverskom inženjerstvu je rezultat postojanja sistema – ne tajni”* (L. Peters)
- Pod *životnim ciklusom* u industriji se podrazumeva inženjerski model rada na projektu koji se sprovodi
- U softverskoj industriji, vlada uverenje da postoji magičan, pravi, nenadmašan, univerzalan tip životnog ciklusa
- Softversko inženjerstvo je veoma srodno inženjerstvu u „klasičnom smislu“, te se koreni različitih modela životnih ciklusa nalaze u praksama starim hiljadama godina
- Nakon ovog kratkog predavanja, mističan i neuhvatiljiv pojam životnog ciklusa razvoja softvera, trebalo bi da bude nešto bliži.

ŽIVOTNI CIKLUS KVALITETA SOFTVERA

(Dva loša ubiše pouzdanost programa)

ŽIVOTNI CIKLUS KVALITETA SOFTVERA

- Najveći napredak na polju poboljšanja kvaliteta i pouzdanosti softvera, učinjen je u okviru softvera sa visokim stepenom dostupnosti
- Životni ciklus kvaliteta softvera – ideja Deutsch-a i Willis-a, 1998.
- Kako softverski sistem postaje prožet kvalitetom?
- Manjkavost, ili greška? Razlika između dva pojma
- Principi životnog ciklusa kvaliteta softvera ne zamenjuju životni ciklus razvoja softvera, već ga upotpunjavaju

ŽIVOTNI CIKLUS KVALITETA SOFTVERA

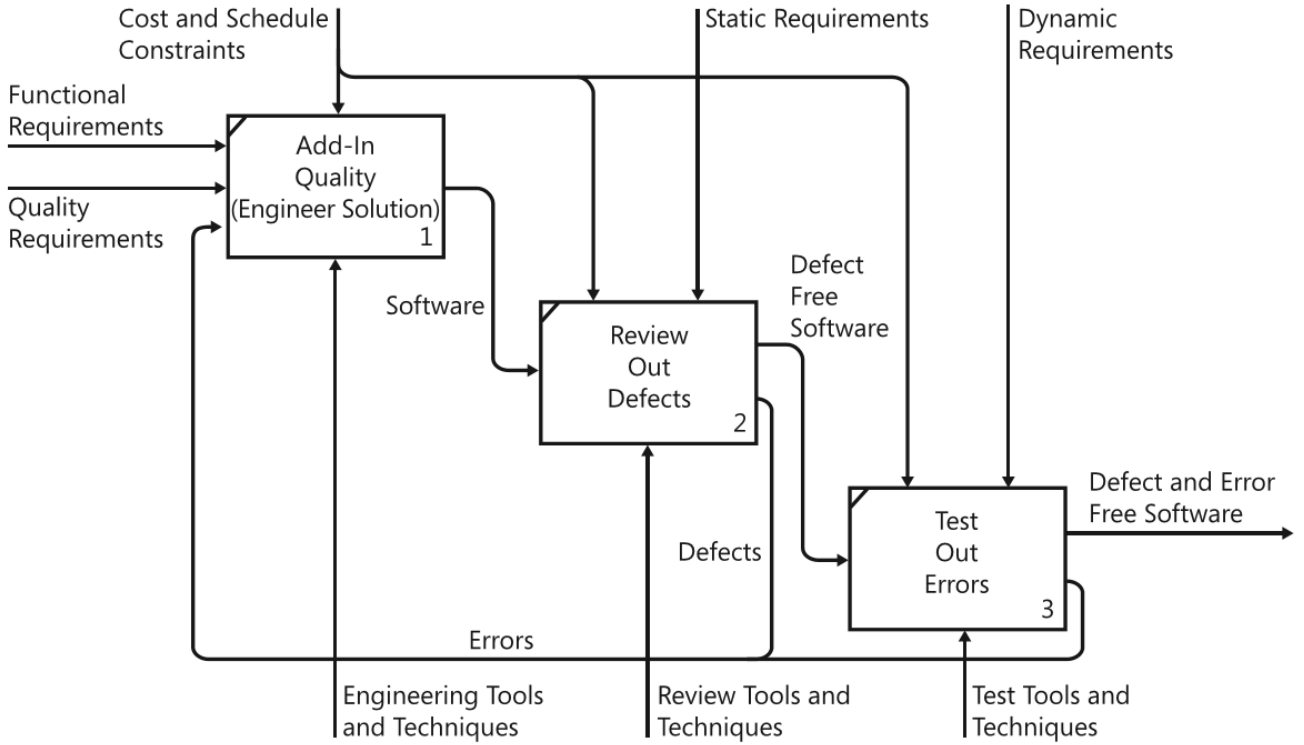

FIGURE 5-1 A detailed view of the software quality lifecycle.

RAZVOJ SOFTVERA POSMATRAN KAO PROCES

(Zato što je Kafkinim romanima potreban veći broj pogodaka u pretraživaču)

RAZVOJ SOFTVERA POSMATRAN KAO PROCES

- Razvoj softvera je u grubim crtama sličan ostalim inženjerski orijentisanim disciplinama
- Za projektnim zahtevom sledi razvoj nacrta, sprovođenja nacrta u konkretan proizvod, testiranje i ispravki stvorenog proizvoda, isporuke proizvoda klijentu.
- Bez obzira na vrstu završnog proizvoda, u svim inženjerskim disciplinama se počinje sa izvesnim problemom ili konceptom, koji je potrebno dovesti do faze sazrevanja – pretvaranja u proizvod ili proces

RAZVOJ SOFTVERA POSMATRAN KAO PROCES

TABLE 5-1 List of Activities in Software Development

Software development phase	Similar engineering phase
Process implementation	Adoption of industry standards
System requirements analysis	Specification
System architecture design	Conceptual design
Software requirements analysis	
Software architecture design	
Software detailed design	Blueprint
Software coding and testing	Implementation
Software integration	
Software qualification testing	Inspection cycle
System integration	
System qualification testing	
System installation	System delivery
System acceptance and support	Punch list completion/final delivery

RAZVOJ SOFTVERA POSMATRAN KAO PROCES

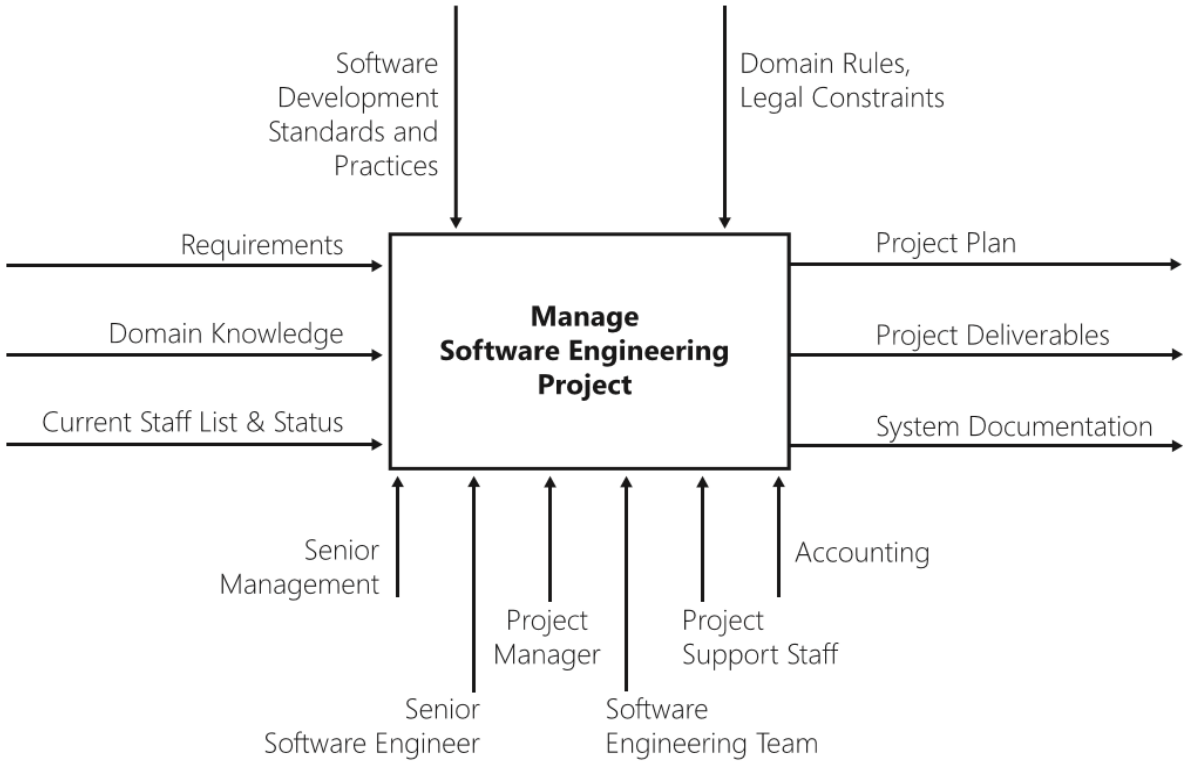

FIGURE 5-2 High-level view of software project management.

RAZVOJ SOFTVERA POSMATRAN KAO PROCES

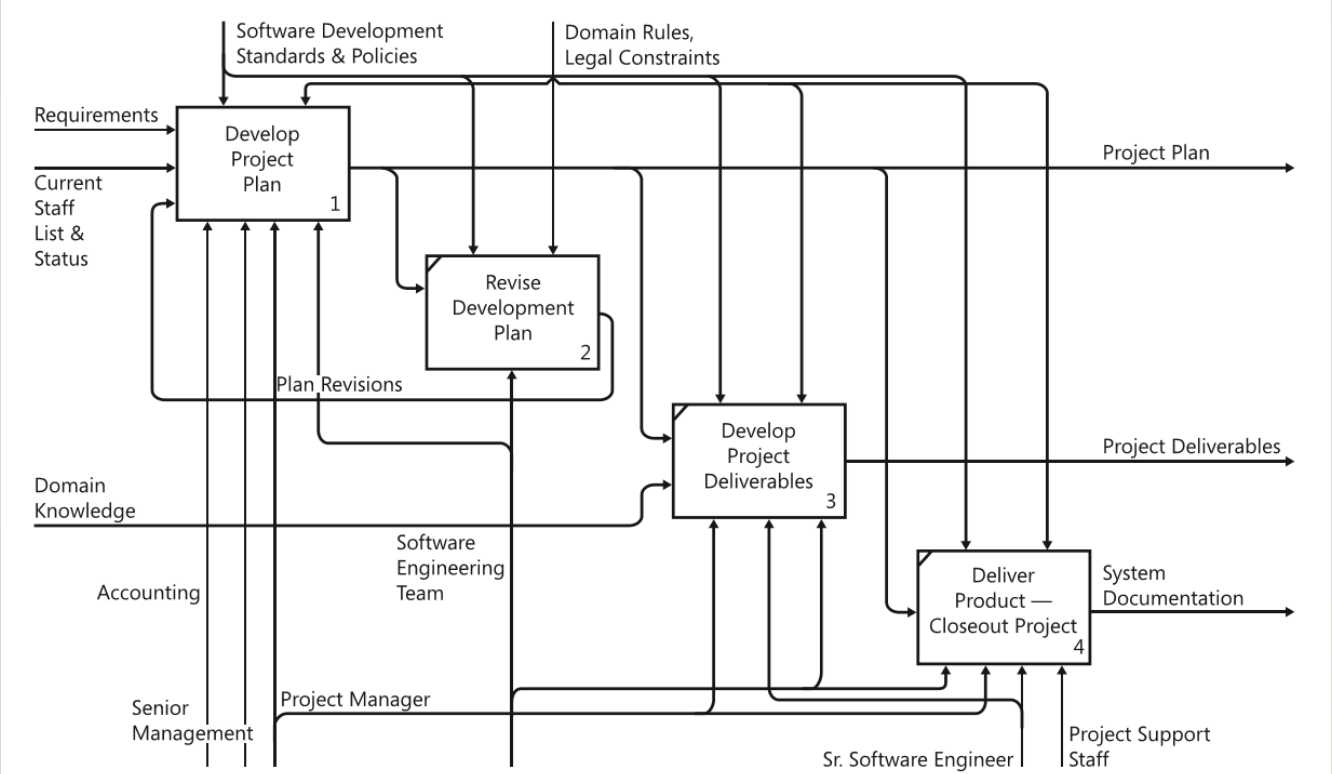

FIGURE 5-3 The four activities of software project management.

RAZVOJ SOFTVERA POSMATRAN KAO PROCES

- Svaka od četiri navedene aktivnosti ima važnu ulogu u procesu razvoja softvera
- Bitan segment četvrte celine – *postpartum* (*postmortem*) sastanci
- Bez obzira da li je u pitanju „pobeda“ ili „poraz“, analiza rezultata je neophodna
- Zamka oslanjanja na ključne osobe, umesto na pouzdane timove
- „Nemamo vremena sada da se bavimo tim problemom, možda kasnije, kada se druge stvari slegnu, ali sada svakako ne“ – samo jedno u nizu samoispunjavajućih proročanstava

RAZVOJ SOFTVERA POSMATRAN KAO PROCES

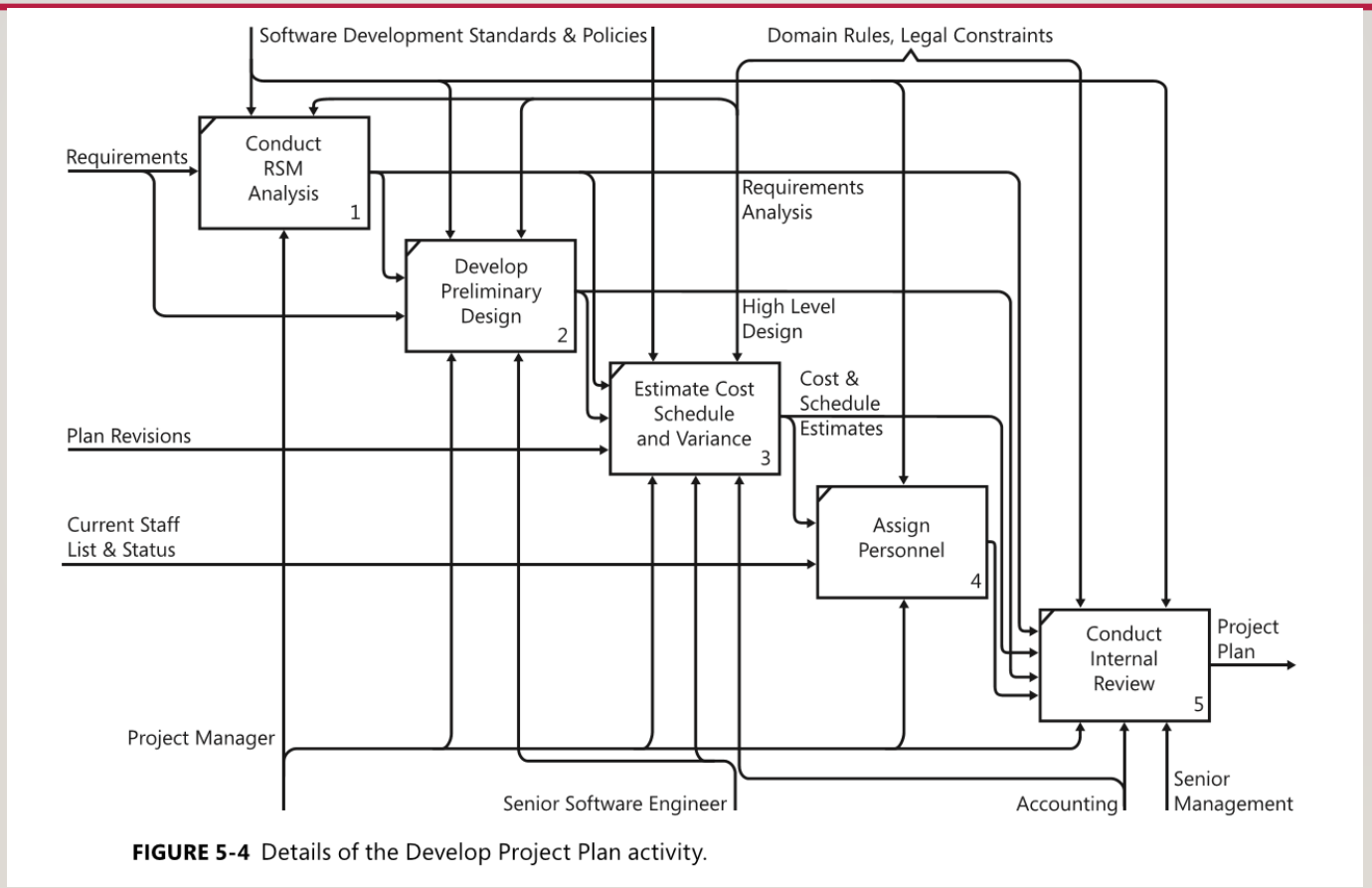

FIGURE 5-4 Details of the Develop Project Plan activity.

MODELOVANJE PROCESA

(Jer procesi moraju biti savršenstvo bez mane)

MODELOVANJE PROCESA

- Procesi su vremenski usklađeni skupovi aktivnosti, koji su usmereni ka cilju ili nekoj vrsti ishoda
- Procesi se ne mogu smatrati apsolutnim algoritmima, receptima za uspeh, već ih treba posmatrati kao smernice
- Razvoj softvera kao delatnost nosi mnogo nepoznanica, te je stoga nemoguće napraviti jasno određen plan koji pri svakoj primeni daje isti rezultat, pa čak ni jednako zadovoljavajući
- Ideja modelovanja procesa, nastala je početkom XX veka
- Primer dobre optimizacije proizvodnog procesa: Fordov „Model T“

MODELOVANJE PROCESA

- Efikasni modeli procesa omogućavaju otkrivanje nekoliko klasa problema:
  - Neefikasnosti u procesu, poput povratnih puteva i nepotrebnog ponavljanja posla
  - Tačaka zagušenja, koje mogu da uspore ili zaustave procese koji od njih zavise
  - Neiskorišćenih mogućnosti za konsolidaciju koraka, koja bi mogla pojednostaviti proces
  - Krupnih previda, poput izostavljanja neophodne aktivnosti (ili skupa aktivnosti)

MODELI ŽIVOTNOG CIKLUSA

(Kako se zove ono što ste već morali da usvojite na prirodan način, ako već niste dobili otkaz)

MODELI ŽIVOTNOG CIKLUSA - OSNOVE

- Ideja svih modela životnog ciklusa jeste da izbegnu ili preduprede pojavljivanje nekog od klasičnih, dobro opisanih teškoća, koje se javljaju u procesu razvoja softvera
- Najčešći problemi: neumereni troškovi, nepostojanje funkcionalnosti, neprihvatanje od korisnika, loša mogućnost održavanja, loša pouzdanost, zakasnela isporuka.
- Nijedan model životnog ciklusa ne predstavlja univerzalno rešenje, iako se tako često predstavlja
- Nekada „ljudski“ deo jednačine može da prevagne, i da i pored lošeg ili nepostojećeg modela, uspe da isporuči dobar rezultat – no ovo je više izuzetak nego pravilo

MODELI ŽIVOTNOG CIKLUSA – OSNOVNI TIPOVI

- Postoje tri osnovna tipa modela životnog ciklusa:
  - Klasični fazni
  - Evolucioni
  - Bezoblični

MODELI ŽIVOTNOG CIKLUSA – KLASIČNI FAZNI

- Klasični fazni
  - Prate tradicionalni pristup razvoju, po ključnim, serijalizovanim fazama
  - Ni naručioci, ni razvijaoci softvera ne vide krajnji proizvod sve do poslednje faze
  - Big bang, odnosno big boom sindrom

MODELI ŽIVOTNOG CIKLUSA – EVOLUCIONI

- Evolucioni
  - Zasnovani na principu da će zahtevi evoluirati u toku razvoja softvera
  - Brzo kreirati svedeni set funkcionalnosti, a onda širiti postepeno opseg rada
  - Neophodan je blizak nadzor od strane upravnika projekta, radi sprečavanja sindroma odbeglog koda
  - Kada je proces gotov?

MODELI ŽIVOTNOG CIKLUSA – EVOLUCIONI

- Bezoblični
  - Malo struktuirani, ili čak potpuno nestruktuirani
  - Oslanjaju se na napore kvalitetnih pojedinaca
  - Teško je uspostavljanje kontrole, koja se uglavnom svodi na postavljanje rokova i limita za završetak projekta
  - „Sreća prati one koji je ne uključuju u svoje planove“ - Zuma

MODELI ŽIVOTNOG CIKLUSA – OPŠTI OBLIK

- Opšti oblik svih modela životnog ciklusa čini napredovanje od problema do rešenja, upotrebom neke sheme zasnovane na postepenom napretku
- Model životnog ciklusa je proces koji upravlja tokom aktivnosti, artefkata, i rezultata, od početka do kraja, u hronološkom redosledu
- Iako je popularnost nekih od navedenih modela životnog ciklusa, od njih praksi stvorila industrijske standarde delovanja, ne postoji tako nešto poput „standardnog“ životnog ciklusa razvoja softvera

MODELI ŽIVOTNOG CIKLUSA – PREGLED

- Model slapa (Waterfall)
- Zakucavanje (Slam dunk)
- Spiralni model
- Evolucioni model
- Etape sa kapijama (Stage gate)
- Brzo prototipovanje (Rapid prototyping)
- Agilno programiranje
- Usklađivanje i stabilizacija (Synchronization and stabilization)

MODELI ŽIVOTNOG CIKLUSA – MODEL SLAPA

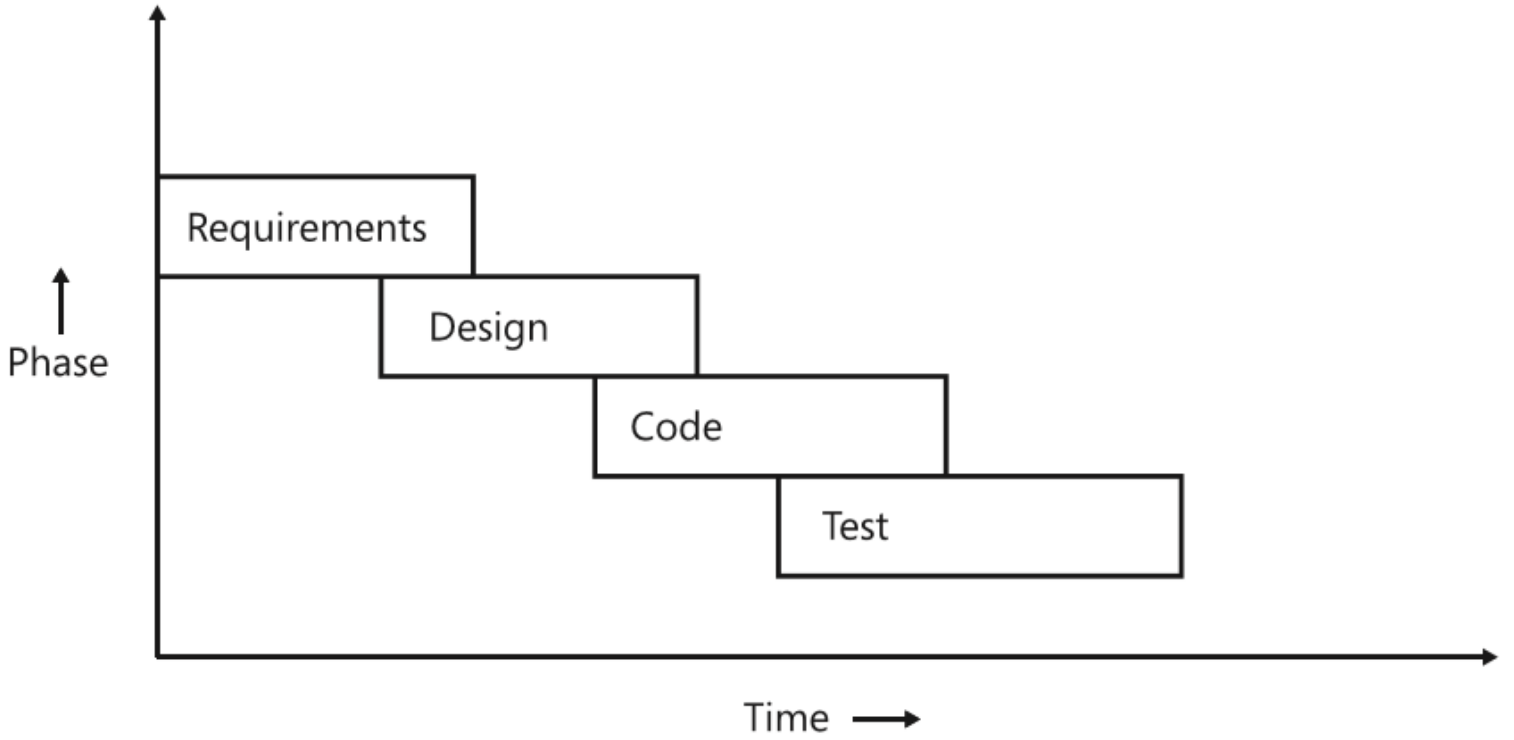

FIGURE 5-7 A representation of the waterfall lifecycle model.

MODELI ŽIVOTNOG CIKLUSA – MODEL SLAPA

- Rezultati iz jedne etape ciklusa se „prelivaju“ u sledeću
- Tek pred kraj celog projekta (u smislu novca i vremena) moguće je videti prve rezultate
- Lako praćenje i dokumentovanje procesa
- Neki eksperti smatraju da je ovaj model čest uzročnik problema
- Postoji mnogo varijacija na temu ovog modela, koje se zasnivaju na uvođenju „podslapova“ i ublažavanju efekta velikog praska

MODELI ŽIVOTNOG CIKLUSA – ZAKUCAVANJE

- Slično kao i u košarci, rešite da date koš, i onda zakucate u velikom stilu i uz gromki aplauz publike.
- Svi učesnici projekta imaju blagu ideju kako bi šta trebalo da izgleda, pristup stvaranju proizvoda se zasniva na pokušaju i grešci
- Smatra se modelom, samo zato što ga neki smatraju mogućim, i što se javlja u praksi
- Uglavnom ne vodi do stvaranja upotrebljivog proizvoda

MODELI ŽIVOTNOG CIKLUSA – SPIRALNI MODEL

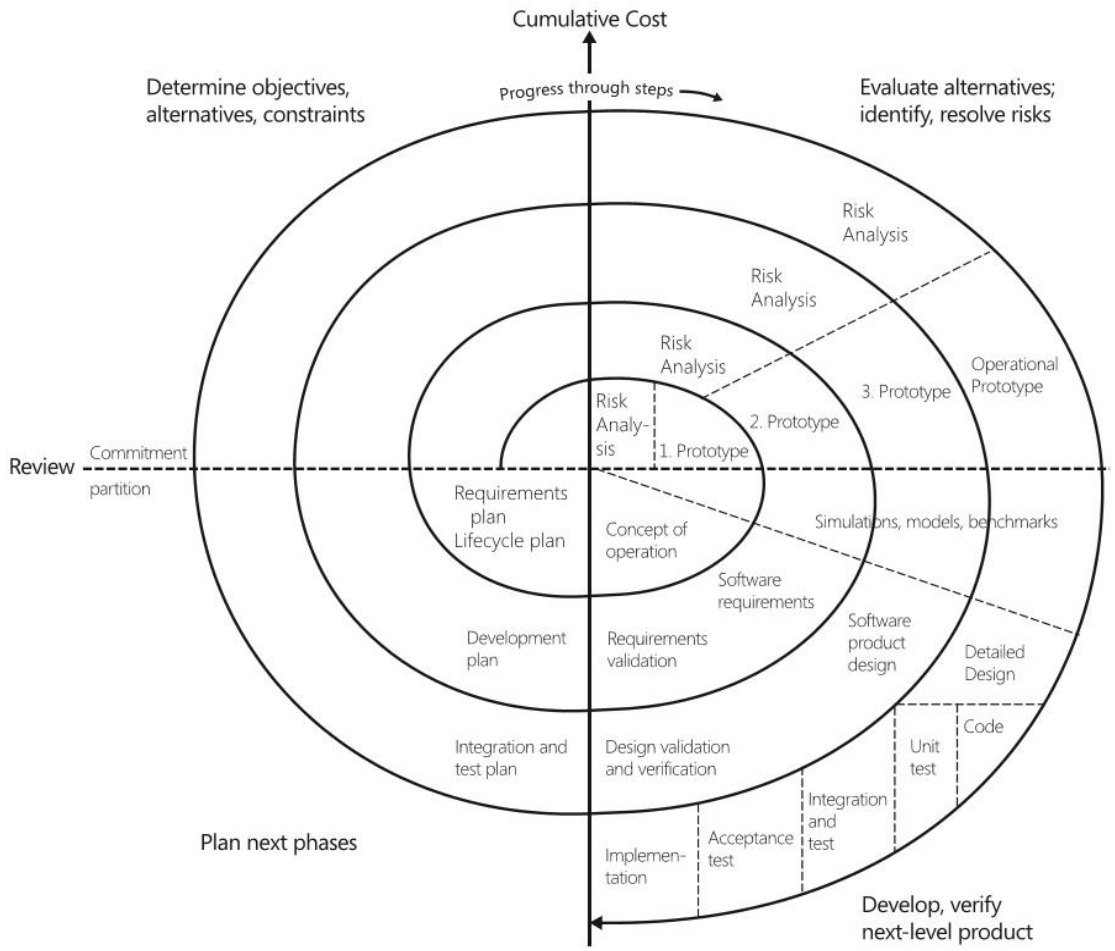

FIGURE 5-8 An example of the spiral lifecycle model.

MODELI ŽIVOTNOG CIKLUSA – SPIRALNI MODEL

- Originalno razvijen od strane Barija Boema (Barry Boehm, 1988)
- Uglavnom prilagođen velikim projektima
- Osnovna ideja je rano otkrivanje i kontrola rizika, tokom celog toka projekta
- Sa svakim prolaskom kroz spiralu, uvećavaju se troškovi, ali se pre prelaska u sledeću fazu pravi procena rizika, kao i operacioni prototip, koji bi u tome trebalo da pomogne
- Implicitno se podrazumeva da, ukoliko je procenjeni rizik pri prelasku u sledeću fazu preveliki, može doći do reorganizacije ili ukidanja projekta

MODELI ŽIVOTNOG CIKLUSA – EVOLUCIONI MODEL

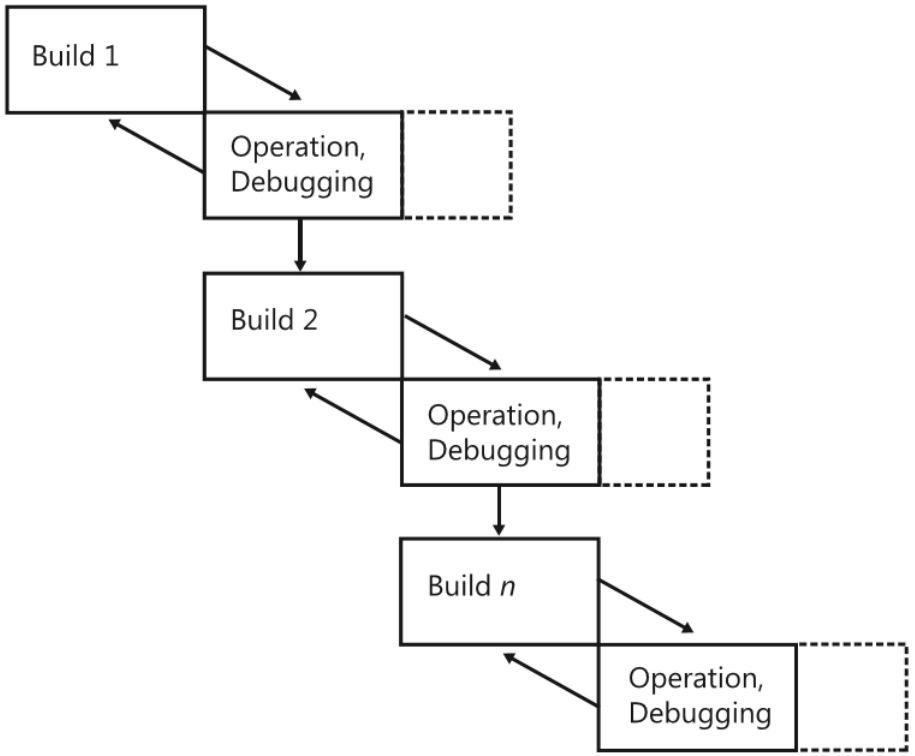

FIGURE 5-9 An example of the evolutionary lifecycle model.

MODELI ŽIVOTNOG CIKLUSA – EVOLUCIONI MODEL

- Ideja evolucionog modela životnog ciklusa je, kao što i ime sugeriše, postepen, evolucioni razvoj proizvoda, od ideje, do uspešne realizacije pune različitih funkcionalnosti, bez mnoštva grešaka
- Malo gradi, malo testiraj, ponovi
- Rafinisana i unapređena verzija ovog modela, poznatija je kao agilno programiranje
- Dobar model, kada je potrebno izgraditi timski duh i kompetencije članova tima koji nemaju mnogo iskustva
- Nekada je dobro kombinovati ovaj model, sa čvrstom arhitekturom važnih i rizičnih celina

MODELI ŽIVOTNOG CIKLUSA – ETAPE SA KAPIJAMA

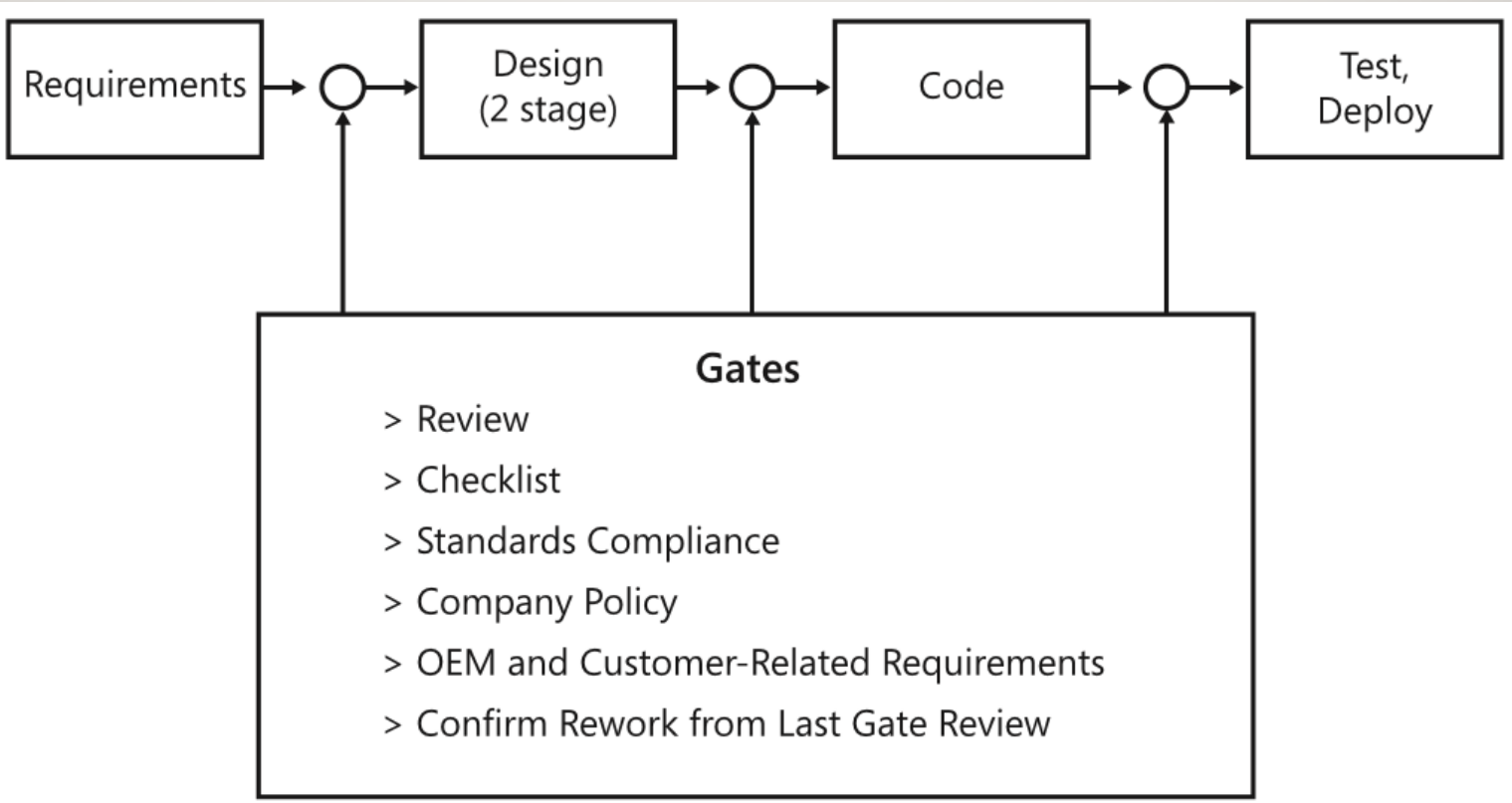

FIGURE 5-10 Phases and gates in the stage gate lifecycle.

MODELI ŽIVOTNOG CIKLUSA – ETAPE SA KAPIJAMA

- Iako na osnovu ilustracije može delovati da je ovakav model u praksi neprimenjiv, zato što su faze u potpunosti serijalizovane, i zahteva se visok stepen pedantnosti izrade, rezultati u praksi su drugačiji
- Dosta posla može biti paralelizovano, a čist i temeljno testiran kod smanjuje vreme potrebno za popravljanje manjkavosti i grešaka u kodu
- Izazovi ovog pristupa: ljudski faktor i potrebna infrastruktura za efikasnu primenu

MODELI ŽIVOTNOG CIKLUSA – BRZO PROTOTIPOVANJE

- Prototip je sredstvo stvaranja razvojne specifikacije
- Fokus ovog pristupa je na klijentu, kojem se u kratkom vremenskom roku obezbeđuje uvid u to kako će proizvod izgledati, kada postane funkcionalan
- Smanjuje se nepoznanica, da li će proizvod biti prihvaćen
- Postoji značajan rizik od stvaranja fenomena odbeglog koda, kao i od usvajanja prototipa kao funkcionalnog proizovoda, zbog nerazumevanja ili nedisciplinovanosti klijenta

MODELI ŽIVOTNOG CIKLUSA – AGILNO PROGRAMIRANJE

- Agilno programiranje nije jedna metodologija, nego je zapravo skup različitih metodologija i dobrih praksi koje vode ka istom cilju – pouzdanom i brzo dostupnom proizvodu, i zadovoljnim klijentima i razvijaocima softvera.
- Jedan je od najdetaljnije opisanih modela, sa mnoštvom literature i primera iz prakse
- Posao se deli na „priče“ od kojih svaka nosi određen broj poena, u zavisnosti od obimnosti i težine
- U svakom ciklusu, procenjuje se koliko koja od osoba može da isporuči poena
- Ukoliko je očigledno da će rokovi biti probijeni, na naručiocu je da odluči da li će se odreći neke funkcionalnosti završnog proizvoda, ili roka isporuke

MODELI ŽIVOTNOG CIKLUSA – AGILNO PROGRAMIRANJE

- Ovaj model podstiče stvaranje poverenja i transparentnost
- Jedna od glavnih mana predstavlja nedostatak stabilnosti – česte promene i dodavanje funkcionalnost onemogućavaju praćenje šire slike
- Kao i svi ostali modeli, nije univerzalno primenjiv

MODELI ŽIVOTNOG CIKLUSA – AGILNO PROGRAMIRANJE

TABLE 5-3 Characteristics of Agile Development

Source	Description
Whole team	Developers, stakeholders, business analysts, quality assurance people all working together as a team.
Planning game	Involves the use of cards or other schemes to identify each feature/ requirement that is part of the project; one feature or requirement per card.
Small releases	Releases occur (typically) on a 2-week cycle. Supported features at each of these internal release points are available for the customer to experience and critique.
Customer tests	Tests are developed by the quality assurance group, one per card. Tests are automated as much as possible except for user interface–related testing, which occurs throughout the project with the biweekly releases.

MODELI ŽIVOTNOG CIKLUSA – AGILNO PROGRAMIRANJE

TABLE 5-4 Developer Practices Used in Agile Development

Term/Attribute	Description
Coding standard(s)	Developed (minimally) at the outset. Evolve over time with participation of entire team.
Sustainable pace	Goal is to develop in small increments, avoiding the death march phenomenon.
Metaphor	The basic concept.
Continuous integration	One of the main tenets of this approach—a working system is built daily.
Collective ownership	All stakeholders are responsible, avoiding scapegoats.

MODELI ŽIVOTNOG CIKLUSA – AGILNO PROGRAMIRANJE

TABLE 5-5 Quality-Related Practices Used in Agile Development

Term/Attribute	Description
Test-driven development	Testing is continuous.
Refactoring	Continual revision of structure, refinement of code, improved quality.
Simple design	Applies the Keep It Simple, Stupid (KISS) principle.
Pair programming	People work in pairs, often one more senior than the other, using one development computer and one keyboard. The advantage is the increase in skill and knowledge of the junior or less-skilled person. The senior person circulates among pairs, expanding the impact of this knowledge sharing.

MODELI ŽIVOTNOG CIKLUSA – USKLAĐIVANJE I STABILIZACIJA

- Ovaj model se koristi u Microsoft-u, kao i, moguće u drugim kompanijama
- Počevši od osnovnih zahteva projekta, tim stvara specifikacije, postavlja prioritete i deli projekat u četiri etape
- Etape: RC1,RC2, RC3, RTM
- U svakoj od narednih etapa, povećava se prag potreban za dozvolu izmene koda, smanjuje se brzina dodavanja novih funkcionalnosti i smanjuje se broj manjkavosti i grešaka
- Glavna ideja ovog modela jeste postizanje stabilnosti i zrelosti konačnog proizvoda

MODELI ŽIVOTNOG CIKLUSA – USKLAĐIVANJE I STABILIZACIJA

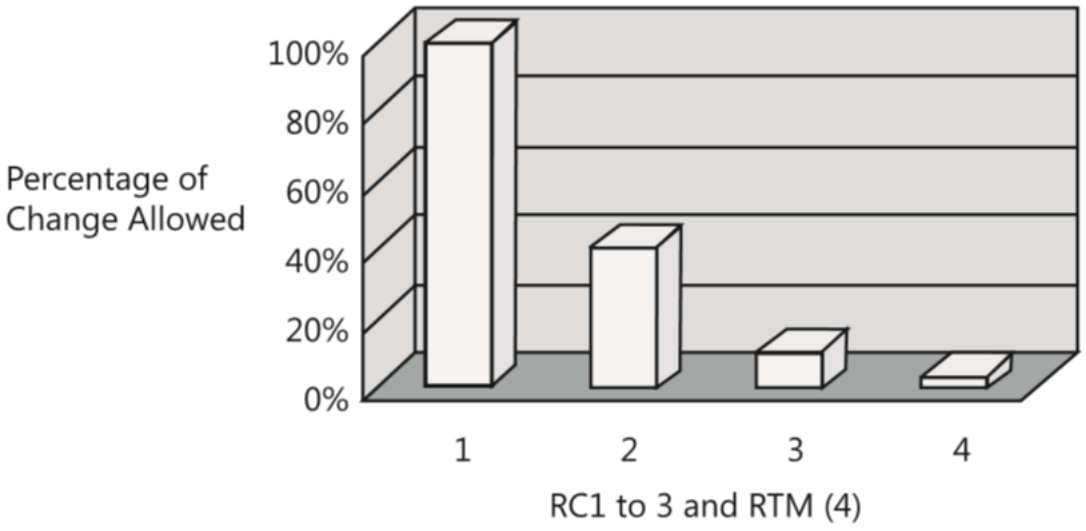

FIGURE 5-13 Change over time in the synchronization and stabilization model.

POREĐENJE OSOBINA RAZLIČITIH MODELA ŽIVOTNOG CIKLUSA

(Ovi mali krompiri i nisu nešto veliki, a ovi veliki, nisu baš preterano mali)

POREĐENJE OSOBINA RAZLIČITIH MODELA ŽIVOTNOG CIKLUSA

- Različiti modeli imaju različite osobine, nijedan nije univerzalno primenjiv
- Neki modeli imaju osobine koje odgovaraju manjim timovima, manje iskusnih ljudi, dok su drugi prilagođeni tako, da omogućavaju dobijanje maksimuma od iskusnih pojedinaca
- Nijedan od modela nije „uklesan u kamenu“ – moguće su i poželjne, modifikacije i kombinacije različitih modela u različitim fazama projekta ili različitim podtimovima

POREĐENJE OSOBINA RAZLIČITIH MODELA ŽIVOTNOG CIKLUSA

TABLE 5-6 Comparison of Common Lifecycle Models

Lifecycle Model	Pros	Cons	Most Applicable To
Waterfall	Controllable results; consistent with engineering practice	Forestalls delivering results until late in the development cycle	All project sizes
Slam dunk	Gets something going right away	Not truly a lifecycle; results generally unsatisfactory	Not recommended
Spiral	Reflects reality of software development in the large; reports of use quite positive	Complex and intricate; requires extensive management infrastructure	Major multimillion-dollar efforts; ill suited to small efforts
Evolutionary	Begin controlled, working software early on; longer testing period than otherwise possible	Nothing stable for long; high change level without much maturity occurring in software	Small to medium projects
Stage gate	Practically guarantees adherence to some discipline	Requires an infrastructure, maturity, and process some firms might not have	Medium to large
Rapid prototyping	Get something going early on; lots of review time	Often results in runaway coding	Small to large
Agile-programming	Puts discipline into the prototyping approach	Requires a mature, disciplined team; strict adherence to process	Small to medium
Synchronization and stabilization	Controls changes; long testing cycle to improve quality	Based on premise that quality can be tested into a system	Small to very large

ODABIR ŽIVOTNOG CIKLUSA RAZVOJA SOFTVERA

(Ako odaberete, bićete nezadovoljni, ako ne odaberete, bićete nezadovoljni)

ODABIR ŽIVOTNOG CIKLUSA RAZVOJA SOFTVERA

- Kao što sami modeli nisu nepromenjivi, tako i odabir modela ne mora biti konačna odluka pri započinjanju projekta – model može biti promenjen više puta, dok se ne nađe oblik koji najviše doprinosi kvalitetnom i pouzdanom radu na projektu
- Neke od ključnih tačaka koje doprinose uspehu:
  - Omogućavanje da karakter projekta bude taj koji diktira koji plan treba da bude upotrebljen
  - Izbegavanje paničnog okretanja modelu Zakucavanje, već odvajanje dovoljno vremena za analizu sledećih koraka
  - Praćenje koliko se dobro izvršava usvojeni plan
  - Postojanje dovoljnog stepena fleksibilnosti, koji dozvoljava neprekidno ponovno planiranje, radi prilagođavanja nepredviđenim okolnostima

ODABIR ŽIVOTNOG CIKLUSA RAZVOJA SOFTVERA

- Pogrešno je misliti da neophodna fluidnost planova znači da planovi nisu potrebni
- Nepostojanje planova zahteva stepen poverenja između naručioca i razvijaoca, koji nije moguć
- Koji god da je model životnog ciklusa odabran, treba ga se disciplinovano držati

ZAKLJUČAK

(Ako je neka dobra duša ostala da sluša izlaganja radova)

ZAKLJUČAK

- Svaki od modela životnog ciklusa predstavlja teoretsku idealizaciju, te stoga treba da bude korišćen kao polazna osnova, a ne kao konačna definicija modela za upotrebu u praksi
- Iskustvo pokazuje da postoje uspešni projekti koji su „radili sve pogrešno“ i neuspešni projekti koji su „radili sve ispravno“ kada je odabri modela životnog ciklusa u pitanju
- Osnov uspešnog rada jeste fleksibilnost koja prati činjenice – a do činjenica se dolazi merenjem, posmatranjem i komunikacijom
- Pitanje odabira modela životnog ciklusa razvoja softvera je zapravo minorno, u poređenju sa važnosti koju imaju prilagodljivost izazovnom i uvek promenljivom tehničkom i poslovnom okruženju, kao i disciplina i posvećenost kvalitetu izrade proizvoda

HVALA VAM NA PAŽNJI

(Ima li pitanja?)