Projektiranje informacijskih sustava

Uvod

Ak. god. 2009/2010

Literatura

- System Analysis and Design, Third Edition; Dennis, Wixom and Roth; Wiley, 2006
- www.wiley.com/college/dennis

2

Informacijski sustav

- Informacijski sustav (*Information System – IS system*) je sustav za prikupljanje, pohranu, obradu i pružanje informacija.
- Premda takav sustav može biti implementiran i bez računala danas se pod pojmom informacijskog sustava podrazumijevaju računalni sustavi za prikupljanje, pohranu, obradu i pružanje informacija.
- IT (*Information Technology*) tehnologija ili (*Information and Communication Technology* - ICT) se bavi korištenjem tehnologije u obradi i upravljanju informacijama (najčešće u velikim organizacijama) osobito korištenjem računala i računalnog softvera za pretvorbu, pohranu, zaštitu, obradu, prijenos i dohvaćanje informacija. Vrlo često se organizacija koja se bavi računalnim tehnologijama i poslovima naziva IT organizacija.
- ITAA (*Information Technology Association of America*) organizacija okuplja američke IT kompanije (Hawlett-Packard, Microsoft, IBM, Oracle, Novell, Yahoo,…..,) www.itaa.org

3

IT pojmovi

- Koja je razlika između informacijskih sustava (*Information Systems*) ili informacijske tehnologije (*Information Technology-IT*), računalne znanosti (*Computer Science-CS*), programskog inženjerstva (*Software Engineering-SE*) i računalnog inženjerstva (*Computer Engineering-CEN*)?
- IT stručnjaci identificiraju potrebu za računalnom tehnologijom dok CEN ili SE stručnjaci razvijaju tu tehnologiju.
- CS stručnjaci bave se više teoretskim dijelom računarstva (algoritmi, strukture podataka, kriptografija …) dok CEN ili SE (ta se dva područja često izjednačavaju) stručnjaci imaju znanja kako teoretske postavke pretvoriti u stvarni softver (programski jezici, mreže, OS, …).

4

System Analysis & Design - SAD

- Pronalaženje što efikasnijih načina izrade sustava
- Svaki sustav u toku izrade prolazi kroz 4 faze razvoja koje čine proces koji se naziva “system development life cycle - SDLC”

5

System Development Life Cycle (SDLC)

- 4 Faze izrade:
  - Planiranje
  - Analiza
  - Dizajn
  - Implementacija
- Svaka faza sastoji se od niza postupaka koji se oslanjaju na definirane tehnike čiji rezultat su dokumenti koji opisuju projekt.

6

Metodologije razvoja informacijskih sustava

- Metodologija je formalno definiran pristup razvoju informacijskog sustava sa definiranim koracima razvoja u kojima se koriste definirani postupci ili tehnike ili metode (npr. use case tehnika za identificiranje zahtjeva), te rezultatima svakog koraka razvoja (npr. prikupljanje zahtjeva rezultira dokumentom sa formalno definiranim zahtjevima) tzv. *deliverables*.
- Koju metodologiju koristiti?
- Ne postoji tzv. “silver bullet” (Frederick Brooks, No Silver Bullet – Essence and Accident in Software Engineering, Information Processing IFIP Proceedings, 1986).
- Metodologija je skupina metoda, postupaka, tehnika i sl. koje se koriste za razvoj informacijskog sustava. Jedna metoda se može koristiti u različitim metodologijama.

7

Razvoj informacijskih sustava

- Metodologije se bave razvojem, implementacijom i održavanjem VELIKIH informacijskih sustava uz ispunjenje osnovnih zahtjeva:
  - na vrijeme,
  - u okviru sredstava,
  - prihvatljive performanse,
  - ispravan, pouzdan, robustan.

8

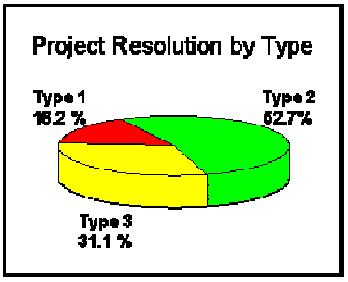
Veliki informacijski sustavi

- više ljudi organiziranih u timove
- pažljiva analiza, dizajn, implementacija i testiranje
- > 100 KLOC (LOC – *line of code*) (1KLOC = 1000 linija koda)
- cijena softvera nadmašuje cijenu hardvera
- sustav je u uporabi više godina i mijenja se s vremenom

9

Razvoj informacijskih sustava

- 1994. godine je Standish Group (http://standishgroup.com/index.php) izvršila analizu 8.380 projekata u državnom i privatnom sektoru u SAD-u i ustanovila:
  - Oko 31% projekata razvoja programske podrške je obustavljeno prije završetka.
  - Oko 53% je kompletirano s prosječnom cijenom koja je za oko 189% premašila predviđenu cijenu, od ovih 53% samo 42% projekata je zadržalo njihove izvorne (originalne) značajke i funkcije.
  - Oko 16% projekata je kompletirano na vrijeme, s predviđenim budžetom i definiranim funkcionalnostima.



10

Razvoj informacijskih sustava

- Podaci Standish Group za 2004. su nešto bolji, ali još uvijek zabrinjavajuće loši.
- Svega 28% projekata može se smatrati uspješnima dok se još uvijek oko 18% započetih projekta napušta.

11

VCF

- VCF (Virtual Case File) case-managment softver za pohranu podataka o slučajevima, za unakrsno pretraživanje podataka i sl. je bio informacijski sustav na kojem su radili inženjeri SAIC-a zajedno sa timom agenata iz FBI-a.
- Projekt je službeno je počeo u listopadu 2001. (iako je modernizacija IT sustava počela malo ranije, u studenom 2000.), a službeno je završen u travnju 2005.
- VCF je trebao zamijeniti postojeći, zastarjeli ACS (Automated Case Support) sustav, objediniti sve dosadašnje baze podataka o zločincima, osumnjičenima, dosjeima, dokazima, te omogućiti pristup iz bilo kojeg ureda FBI-a svim informacijama i arhivi postojećih dosjea, dokaza i sl., dakle pružiti mogućnost agentima da dijele informacije.
- Gotovo ništa od predviđenih ciljeva nije bilo ispunjeno, a potrošeno je mnogo vremena i 581 milijun dolara.

12

Zašto SAD?

- Cijena pogreške može biti velika:
  - bankrot proizvođača softvera
  - bankrot klijenata koji ovise o softveru
  - financijski gubici zbog grešaka u softveru (*bugova*), nefleksibilnosti sustava ili vremena "nerada" sustava (*downtime*)
  - ljudski gubici u sigurnosno kritičnim sustavima
- Profitabilnost softvera ovisi znatno o troškovima održavanja i *upgrade* softvera
  - zahtjevi za promjenjivim dizajnom
  - modularnost, evolucija, portabilnost, odvajanje podataka od funkcionalnosti

13

Planiranje

- Osnovni proces shvaćanja zašto se izrađuje sustav i kako će projektni tim razvijati sustav.
- Sastoji se od dva koraka:
  1. Pokretanje projekta (*project initiation*) je faza u kojoj se procjenjuje poslovna vrijednost sustava (koliko će smanjiti troškove ili povećati prihode kompanije). Dobiveni dokumenti su studija provedivosti (*feasibility analysis*) i zahtjev sustava (*system request*).
  2. Upravljanje projektom (*project management*) je faza u kojoj se izrađuje radni plan (*workplan*), određuje raspodjela posla te definiraju tehnike izrade, kako bi se pomoglo projektnom timu u kontroliranju i usmjeravanju projekta kroz cijeli ciklus SDLC-a. Dobiveni dokument je projektni plan (*project plan*).

14

Analiza

- Analiza sustava treba dati odgovore na pitanja tko će biti korisnik sustava, što će sustav raditi, i gdje i kada će se koristiti (dobro mjesto za korištenje dijagrama slučajeva korištenja).
- U ovoj fazi trebalo bi proučiti postojeće sustave (vrlo rijetko razvoj nekog informacijskog sustava kreće od nule), odrediti što se može unaprijediti te postaviti osnovni koncept sustava.

15

Analiza

- Sastoji se od tri koraka:

1. Analiza strategije uključuje analizu već postojećeg sustava (*as-is system*), a služi usmjeravanju projektnog tima u izradi novog sustava (*to-be system*).
2. Prikupljanje zahtjeva (*requirements gathering*) služi u svrhu izrade koncepta za novi sustav, koji će služiti kao predložak za izradu modela za analizu.
3. Prijedlog sustava (*system proposal*) se predstavlja sponzoru projekta i drugim ključnim osobama koje odlučuju o tome da li se izrada sustava nastavlja ili ne. To je prvi dokument koji opisuje koje poslovne zahtjeve sustav treba ispunjavati.

16

Dizajn

- Rezultat dizajna sustava treba biti odluka o načinu funkcioniranja sustava u smislu određivanja:
  - potrebnog hardvera, softvera, mrežne infrastrukture;
  - korisničkog sučelja, formi i izvještaja koji će se koristiti u sustavu;
  - specifičnih programa, baza i datoteka koje će biti potrebne.
- Većina strateški odluka u vezi informacijskog sustava se donosi u fazi dizajna (web aplikacija ili desktop aplikacija, windows ili linux OS,…).

17

Dizajn

- Sastoji se od četiri koraka:

1. Strategija dizajna (*Design Strategy*): Strategija dizajna sustava određuje hoće li se željeni sustavu razvijati unutar organizacije, hoće li se unajmiti neka druga kompanija za razvoj sustava ili će se kupiti već gotovi programski paket.
2. Dizajn arhitekture (*Architecture Design*): Određuje hardver, softver i mrežnu infrastrukturu koja će se koristiti. Često se radi o proširenju i pojačanju postojeće arhitekture.
3. Specifikacija baza i datoteka (*Database and File Specifications*): Definiraju su točno podaci koji će se pohranjivati i gdje će se pohranjivati.
4. Dizajn programa (*Program Design*): Definira koje programe je potrebno napraviti i što će ti programi raditi.

18

Implementacija

- U fazi implementacije sustav se ili razvija ili se kupuje (ukoliko je strategijom dizajna definirano da je bolje kupiti postojeći softverski sustav).
- Ova faza je obično najduža i zahtjeva najviše financijskih sredstava.

19

Implementacija

1. Izrada sustava (*System Construction*): U ovom koraku sustav se izrađuje i testira kako bi se osiguralo da se ponaša u skladu sa zahtjevima.
2. Instalacija (*Installation*): U postupku instalacije sustav se stavlja u rad. Ukoliko je postojala starija verzija sustava (čest slučaj) stari sustav se može ugasiti prije no što se novi stavi u funkciju, ili novi sustav može neko vrijeme raditi paralelno sa starim zbog dodatnog testiranja ili se novi sustav uvodi dio po dio, a ne sve odjednom. Ova faza uključuje i obuku korisnika.
3. Plan podrške (*Support Plan*): Definira post-implementacijsku reviziju sustava (*upgrade* i sl.).

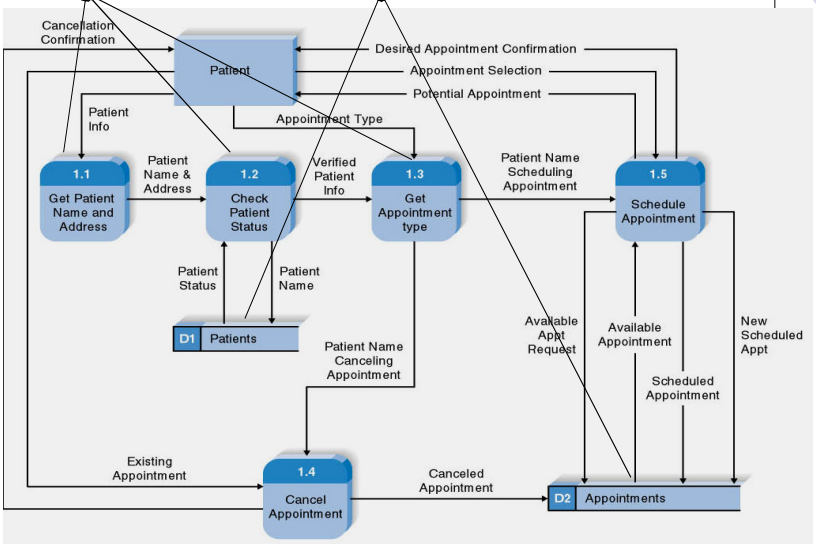
20

Metodologije razvoja sustava

- Već smo naglasili da metodologije pružaju formalizirani pristup implementaciji SDLC-a.
- Metodologije razvoja mogu se podijeliti na:
  - usmjerene na proces koji se informacijskim sustavom obuhvaća (sustav za obračunavanje plaća: proračun radnih sati za mjesec, obračun prethodnih plaća, izračun plaće za ovaj mjesec,….)
  - usmjerene na podatke koji su obuhvaćeni sustavom (sustav za obračunavanje plaća: podaci o radnim satima, podaci o poreznim olakšicama, ….)
  - objektno-orijentirane metodologije obično balansiraju između procesno orijentiranog i podatkovno orijentiranog pristupa informacijskom sustavu

21

Procesi ↔ podaci



22

Metodologije razvoja sustava

- Metodologije s naglaskom na proces:
  - Bitno je fokusirati se na definiranje aktivnosti povezanih sa sustavom
  - Cilj je prikazati sustav kao skup procesa s informacijom koja ulazi i izlazi iz njih (tok informacija = flow)

23

Metodologije razvoja sustava

- Metodologije s naglaskom na podatke:
  - Bitno je definirati sadržaj spremnika podataka i način na koji su oni organizirani.
  - Koriste se modeli podataka kao osnova formalnog opisa sustava.

24

Metodologije razvoja sustava

- Objektno orijentirane metodologije:
  - Balans između prethodne dvije metodologije (proces ↔ podaci)
  - Za opis sustava najčešće se koriste UML dijagrami
  - Sustav se opisuje kao skup objekta koji uključuju i procese i podatke.

25

Metodologije razvoja sustava

- Podjela metodologija prema kategorijama metodologija s obzirom na formalne zahtjeve, brzinu i složenost razvoja sustava:

1. Metodologije strukturiranog dizajna (*Structured design*) (1980-tih) uvode formalne modele sustava, najčešće u obliku dijagrama. Prednosti su u pouzdanosti zbog strogih formalnih zahtjeva, a nedostatci u sporosti i složenosti.
2. RAD (*Rapid Application Development*) metodologije (1990-te) ubrzavaju razvoj sustava. Mogu kao i metodologije strukturiranog dizajna biti ili procesno orijentirane ili podatkovno orijentirane ili objektno-orijentirane.
3. Metodologije agilnog razvoja (*Agile Development*) još više ubrzavaju razvoj sustava, ali na štetu formalnih zahtjeva. Ovo je najnovija kategorija metodologija i još se razvija.

26

STRUKTURIRANI DIZAJN

- Primjenjuje se formalni “korak po korak” pristup SDLC-u, faze se strogo odvojene, a sustav se razvija korak po korak tj. faza po faza
- Uvodi se korištenje formalnog modeliranja ili tehnika koje koriste dijagrame da bi se opisali glavni poslovni procesi sustava

27

Metodologija Vodopada (*waterfall development*)

- Analitičari (razvojni tim) i korisnici sekvencijalno razvijaju fazu po fazu projekta
- PREDNOSTI
  - Zahtjevi sustava su definirani dugo prije nego počne samo programiranje
  - Promjene zahtjeva su svedene na minimum kako se projekt razvija

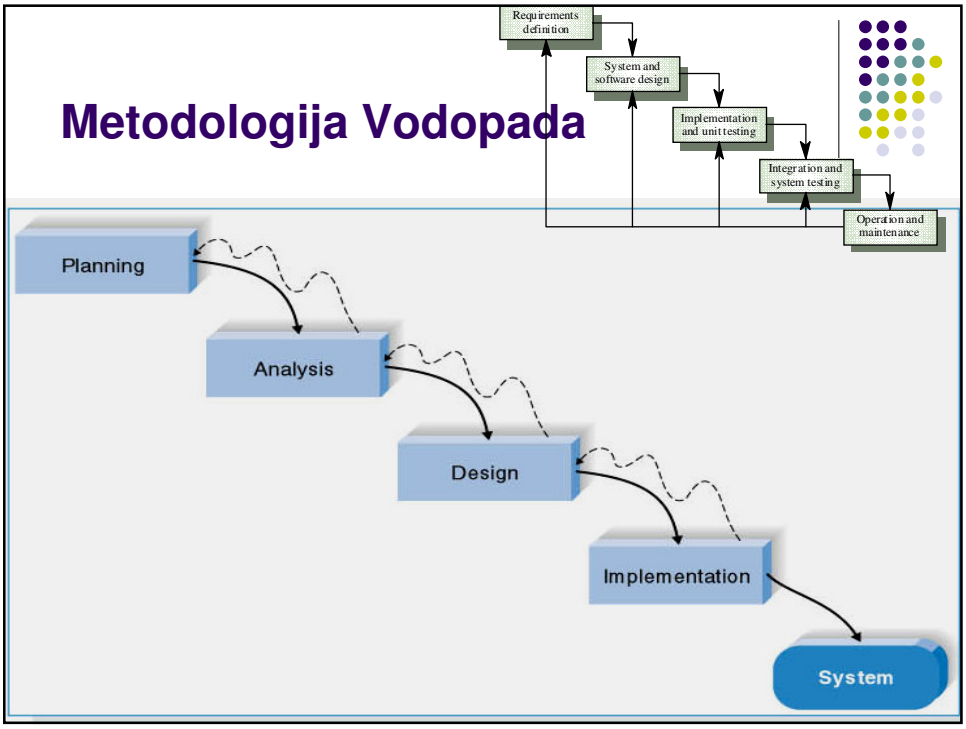
28

Metodologija Vodopada

- NEDOSTACI
  - Dizajn mora kompletno biti specificiran prije nego počne programiranje
  - Predugo vremena protekne između završetka prijedloga sustava u fazi analize i same isporuke sustava

29

Metodologija Vodopada



Metodologija Paralelnog razvoja (*parallel development*)

- Cilj je smanjiti vremenski interval koji protekne između analize i isporuke sustava kod vodopadnog pristupa.
- Umjesto da se cijeli sustav izrađuje sekvencijalno (dio po dio), sustav se u fazi dizajna dijeli na više podsustava koji se paralelno izrađuju i integriraju.
- Na kraju se vrši finalna integracija svih podsustava te se vrši isporuka.

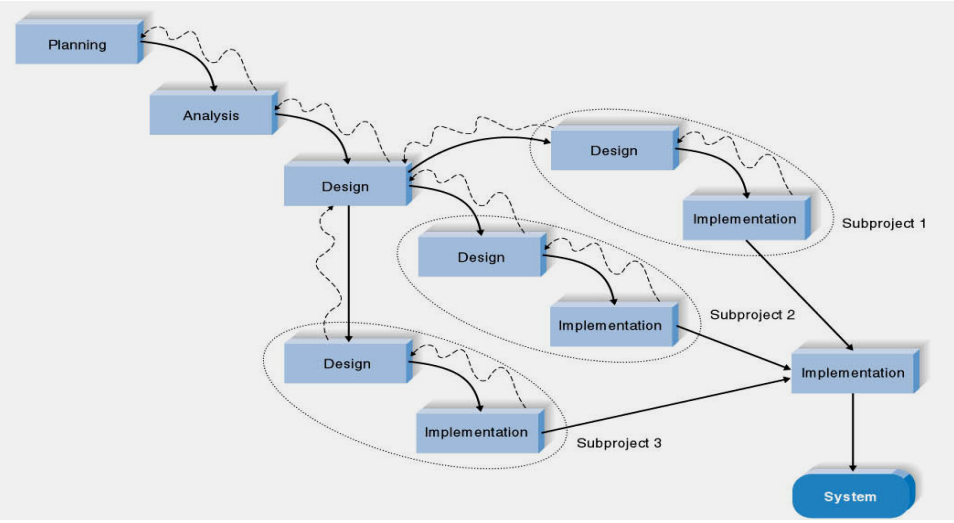
31

Metodologija Paralelnog razvoja

- PREDNOSTI
  - Vrijeme koje protekne do isporuke se smanjuje, što dovodi do toga da je vjerojatnost promjene početnih zahtjeva zbog promjene u stvarnom poslovanju kompanije smanjena
- NEDOSTACI
  - Ako podsustavi nisu samostalni, dolazi do problema u integraciji ako planiranje nije dobro izvršeno

32

Metodologija Paralelnog razvoja



BRZI RAZVOJ APLIKACIJE
(Rapid Application Development - RAD)

- Metodologija se razvija ’90-ih kao rješenje nedostataka strukturiranog razvoja.
- SDLC faze se prilagođavaju kako bi se što prije dijelovi sustava razvili i dostavili kupcu.
- Oslanjaju se na korištenje CASE (Computer-Aided Software Engineering) i sličnih alata koji ubrzavaju proces izrade, viših programskih jezika sa mogućnošću vizualnog razvoja, alata za automatsko generiranje kôda iz specifikacija iz faze dizajna, itd.
- Mogući nedostatak je što se korisnički zahtjevi mogu mijenjati tokom izrade pošto korisnici koriste sustav dok je još u izradi.

34

CASE

- *Computer-Aided Software Engineering* (CASE) obuhvaća različite programske alate za podršku aktivnosti u softverskom procesu kao što su:
  - analiza zahtjeva (*requirements analysis*),
  - modeliranje sustava (*system modeling*),
  - testiranje (*debugging and testing*)…
- Sve metodologije se danas koriste CASE alatima kao što su različiti uređivači, moduli za analizu i provjeravanje modela sustava u suglasju s postavljenim pravilima, kao i generatori izvještaja koji pomažu u izgradnji sustava dokumentacije.
- CASE alati također uključuju i generatore izvornog programskog kôda.
- CASE alat koji podržavaju analizu i oblikovanje se ponekad nazivaju CASE alati visoke razine, jer u suštini podržavaju rane faze u razvoju softvera.
- CASE alate za podržavanje implementacije i testiranja (*debuggers*, *program analysis systems*, *test case generators*, *program editors*) često nazivamo CASE alatima niske razine.

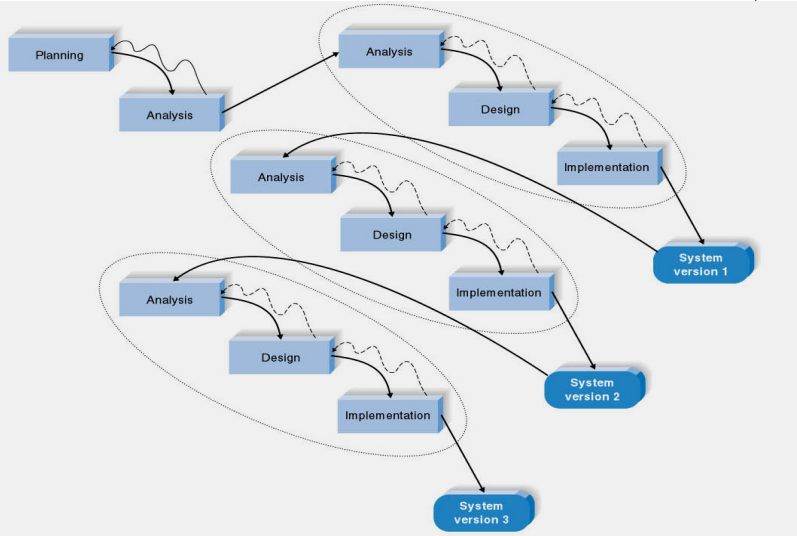
35

Metodologija razbijanja u faze (*phased development*)

- Cijeli sustav se “razbija” u seriju verzija koje se razvijaju sekvencijalno.
- Zahtjevi se kategoriziraju kroz verzije, a najvažniji i neophodni su uključeni u prvu verziju sustava.
- Nakon toga faza analize vodi u dizajniranje i implementaciju, ali samo prve verzije.
- Nakon što je svaka verzija sustava završena, tim odmah počinje rad na slijedećoj.
- Nedostatak je što je izuzetno bitno točno odrediti najvažnije zahtjeve koji moraju biti uključeni u prvu verziju sustava i ako se tu pogriješi cijeli sustava će vjerojatno biti loš.

36

Metodologija razbijanja u faze



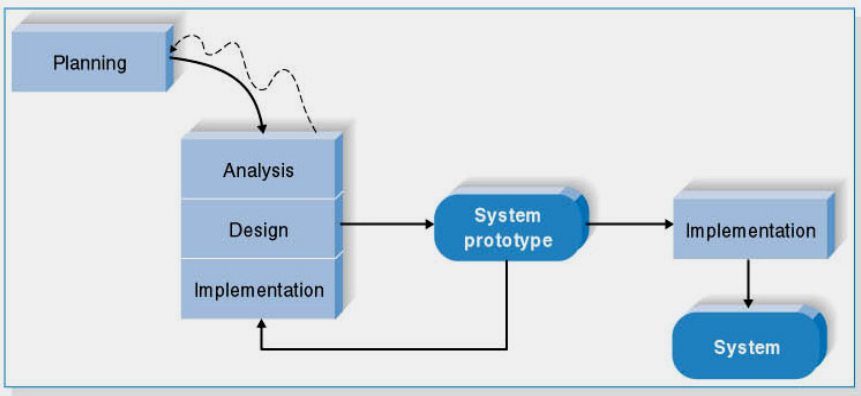
37

Metodologija izrade prototipa (*prototyping*)

- Istodobno se izvode faze analize, dizajna i implementacije, i to postepeno sve dok izrada sustava ne završi.
- Prototip je mala verzija sustava s minimalnim setom funkcionalnosti koja se odmah počne razvijati da bi se sustav što prije isporučio korisniku.
- Prednost je što korisnici imaju uvid u stanje izrade sustava (nema dugih perioda izrade i čekanja).
- Problem ove metode je u tome što se nakon određenog vremena mogu uočiti propusti nastali u prvim verzijama jer analiza i dizajn nisu detaljni kako kod faznog razvoja. Prvi prototipovi su temelj svih budućih verzija sustava te njihova promjena rezultira velikim utroškom vremena.

38

Metodologija izrade prototipa



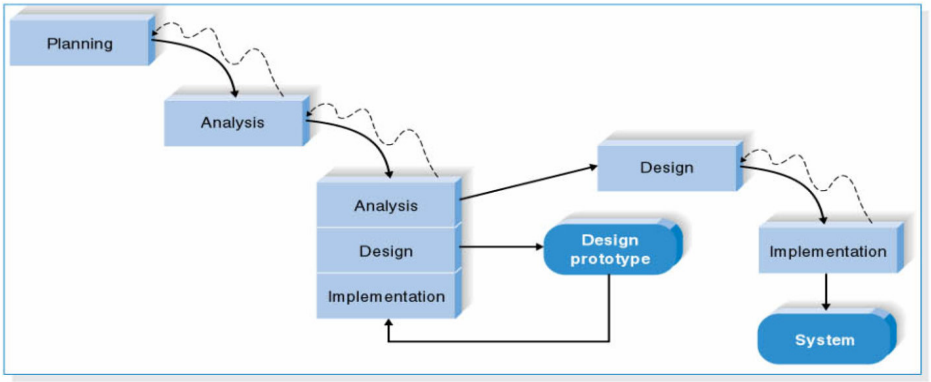
39

Metodologija odbačenih prototipova

- Metodologija odbačenih prototipova (*throwaway prototyping*) je slična prethodnoj metodologiji, samo što se sada prototip koristi samo kao sredstvo za osiguravanje ispunjavanja zahtjeva korisnika (nakon toga se "baca") i to obično samo za dijelove sustava za koje korisnički zahtjevi nisu do kraja identificirani pa napravljeni prototip olakšava korisniku definiranje zahtjeva.
- Korisnik dobiva na uvid prototip dizajna sustav, te nakon što se on odobri kreće se u izradu samog sustava, na osnovu tog prototipa, ali sam prototip se odbacuje.
- Ova metodologija produžuje vrijeme izrade u odnosu na prethodnu, ali zato u pravilu rezultira stabilnijim i pouzdanijim sustavom.

40

Metodologija odbačenih prototipova



41

AGILNI RAZVOJ

- Cilj je postići što brži i efikasniji razvoj aplikacije.
- Odbacuje se dio modeliranja i dokumentacije, a uvodi se jednostavni iterativni razvoj.
- Neke od glavnih metoda su:
  - Ekstremno programiranje (*Extreme programming*-XP)
  - Scrum
  - Dynamic Systems Development Method (DSDM)

42

Ekstremno programiranje

- Bazira se na:
  - Komunikaciji (između razvojnog tima i korisnika, te između članova razvojnog tipa)
  - Jednostavnosti (koristi se KISS (Kip It Simple, Stupid) princip u razvoju aplikacije, koristi se postupak refaktoriranja kôda (code refactoring) kojim se mijenja izvorni kôd bez promjene funkcionalnosti kôda nego samo promjene nefunkcionalnih karakteristika kôda (npr. složenost kôda))
  - Povratnoj informaciji (*feedback*) (korisnici su kontinuirano uključeni u razvoj sustava, te u svakom koraku daju povratnu informaciju timu o dodatnim zahtjevima i sl.)

43

Ekstremno programiranje

- Glavni principi u izradi uspješnog sustava
  - Kontinuirano testiranje
  - Što jednostavnije programiranje, u parovima
  - Što bolja interakcija s korisnicima (ponekad su dio razvojnog tima)
- Nakon površnog procesa planiranja, projektni tim iterativno provodi faze analize, dizajna te na kraju implementacije sustava.

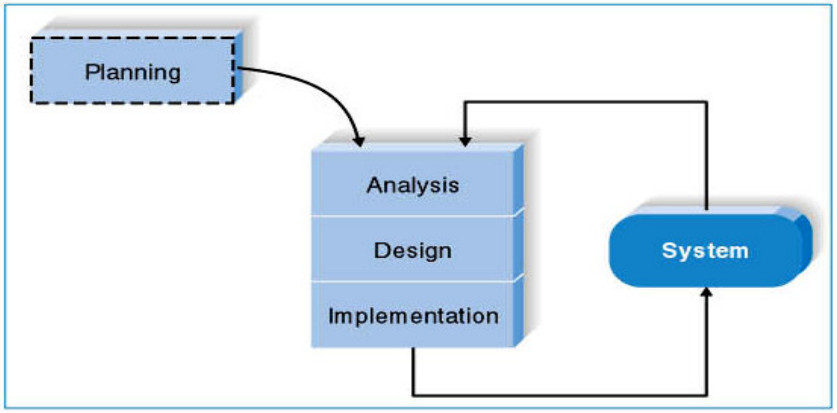
44

Ekstremno programiranje

- Namijenjeno za timove do 10 članova uz uvjete:
  - Visoka motivacija
  - Osjećaj zajedništva
  - Stabilnost
  - Iskustvo
- Najbolje koristiti za izradu manjih i srednje velikih aplikacija, zbog nedostatka dokumentacije.

45

Ekstremno programiranje



46

Izbor odgovarajuće metodologije razvoja

- S obzirom na broj i raznolikost pristupa, izbor nije lak. Nijedna metodologija nije najbolja u svakom slučaju i svaka ima određene prednosti i nedostatke.
- U praksi kompanije obično imaju:
  - Jednu uhodanu metodu
  - Više metoda koje se koriste po potrebi
  - Ništa od navedenog
- Postoje kriteriji za odabir metodologije.

47

Izbor odgovarajuće metodologije razvoja

Ability to Develop Systems	Structured Methodologies		RAD Methodologies			Agile Methodologies
	Waterfall	Parallel	Phased	Prototyping	Throwaway Prototyping	XP
with Unclear User Requirements	Poor	Poor	Good	Excellent	Excellent	Excellent
with Unfamiliar Technology	Poor	Poor	Good	Poor	Excellent	Poor
that are Complex	Good	Good	Good	Poor	Excellent	Poor
that are Reliable	Good	Good	Good	Poor	Excellent	Good
with a Short Time Schedule	Poor	Good	Excellent	Excellent	Good	Excellent
with Schedule Visibility	Poor	Poor	Excellent	Excellent	Good	Good

48

Definiranost korisničkih zahtjeva

- Ako zahtjevi nisu jasno definirani, ne može ih se lako protumačiti niti objasniti programerima.
- Korisnici moraju znati kako tehnologija funkcionira da znaju što mogu od sustava očekivati.
- RAD metode s prototipovima te agilne metode (uz prisutnost korisnika u projektnom timu) su najbolje rješenje ovog problema.

49

Poznavanje tehnologije

- Ako se koristi nova tehnologija pri razvoju moguće je da se zbog nepoznavanja ili krivog korištenja naiđe na neplanirane probleme (npr. prije ste radili u ASP.NET web aplikacije, a sada se trebate prebaciti na JSP).
- Moguće je da odabrana tehnologija nema mogućnosti izvršiti zahtjeve za koje se očekivalo da može.
- Pogodna metoda je metoda odbačenih prototipova jer omogućava i upoznavanje sa tehnologijom i provjeru mogućnosti tehnologije. Također i metoda faznog razvoja (tj. razbijanja na faze) jer se tehnologija upozna prije no što se dođe do konačne verzije sustava.
- Zašto prototipovi baš i nisu najbolja metoda u ovom slučaju?

50

Kompleksnost sustava

- Kompleksni sustavi zahtijevaju opreznu i detaljnu analizu i dizajn, pogotovo ako njihovo loše funkcioniranje može rezultirati velikom štetom bilo kakvog oblika.
- Projektni timovi koji slijede fazni razvoj manje pozornosti pridodaju analizi cijelog problema.
- Iz tog razloga se za ovakve sustave predlaže model odbačenih prototipova.

51

Pouzdanost sustava

- Sustavi za npr. medicinsku opremu ili nuklearne elektrane na prvom mjestu moraju biti pouzdani, dok nekim sustavima poput igara to ne mora biti važna karakteristika.
- Za dizajniranje pouzdanih sustava predlaže se korištenje throwaway prototipa (ne prototipa sustava) zbog toga što se na taj način vrši dvostruka kontrola, prvo zahtjeva, a onda se programiranje vrši odvojeno, s već gotovim predloškom u obliku prototipa.

52

Transparentnost toka izrade

- Vrlo je teško predvidjeti tok izrade sustava, pogotovo neiskusnim timovima.
- Metode RAD-a pomažu u prepoznavanju i uklanjanju rizičnih faktora, kako bi se razvoj odvijao u pretpostavljenim rokovima.

53

Kratki vremenski rokovi

- Za sustave s kratkim rokom isporuke koriste se metode agilnog razvoja, jer je njihova osnovna namjena ubrzavanje razvoja.
- Također se koriste i RAD metode prototipa sustava i faznog razvoja, jer se sustav može isporučiti brzo s djelomičnom funkcionalnošću.

54

SPOSOBNOSTI I ULOGE ČLANOVA PROJEKTNOG TIMA

- Kako bi projektni tim bio uspješan, mora se sastojati od skupa iskusnih stručnjaka iz različitih područja
- Sposobnosti članova se mogu podijeliti na:
  - Tehničke,
  - Poslovne,
  - Analitičke,
  - Međuljudske,
  - Organizacijske….

55

SPOSOBNOSTI I ULOGE ČLANOVA PROJEKTNOG TIMA

- Unutar tima, podjela na uloge je slijedeća:
  - Poslovni analitičar
  - Sistem analitičar
  - Infrastrukturni analitičar
  - Analitičar promjena (Change Management)
  - Voditelj ili menadžer projekta

56

SPOSOBNOSTI I ULOGE ČLANOVA PROJEKTNOG TIMA

Role	Responsibilities
Business analyst	Analyzing the key business aspects of the system
	Identifying how the system will provide business value
	Designing the new business processes and policies
Systems analyst	Identifying how technology can improve business processes
	Designing the new business processes
	Designing the information system
	Ensuring that the system conforms to information systems standards
Infrastructure analyst	Ensuring the system conforms to infrastructure standards
	Identifying infrastructure changes needed to support the system
Change management analyst	Developing and executing a change management plan
	Developing and executing a user training plan
Project manager	Managing the team of analysts, programmers, technical writers, and other specialists
	Developing and monitoring the project plan
	Assigning resources
	Serving as the primary point of contact for the project

57