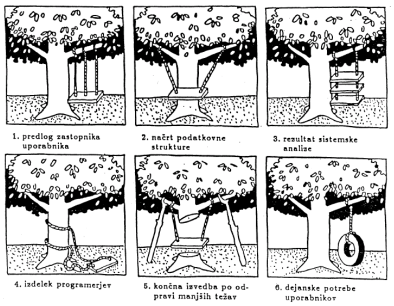
Dokumentiranje informacijske tehnologije

Peter Peer

peter.peer@fri.uni-lj.si
http://www.lrv.fri.uni-lj.si/~peterp

v0.9 9/2003

Problemi razvoja sistemov #1



Problemi razvoja sistemov #2

V Fortranskem programu, ki je kontroliral vesoljsko sondo Mariner na poti k Veneri, je zgolj zamenjava vejice s piko povzročila njeno izgubo. Namesto `DO 3 I = 1,3` je v programu pisalo `DO 3 I = 1.3` in prevajalnik je namesto zanke, ki naj bi se trikrat izvedla, pripisal spremenljivki `DO3I` vrednost `1.3`.

Izhodišče za pripravo snovi

Zloženka & Katalog znanj – Informatika – VSŠ Velenje –

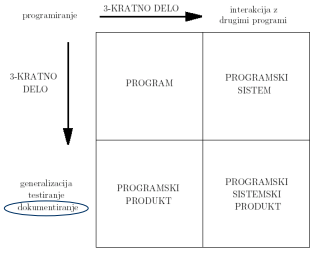
Predmetnik & tipična dela:

- Modul: Programske aplikacije in pod. baze
  1. **Načrtuje, pripravi, izvede in kontrolira lastno delo** in delo skupine
  6. **Vzdržuje dokumentacijo programskih rešitev in pripravlja uporabniška navodila**
- Modul: Računalniški sistemi in omrežja
  1. **Analizira obstoječe stanje, pripravlja rešitve za izboljšanje delovanja RSO ter sodeluje pri njihovi izvedbi**
  6. **Zbira, vodi, arhivira dokumentacijo o celotnem sistemu, delu, postopkih in nalogah**

Oris predmeta

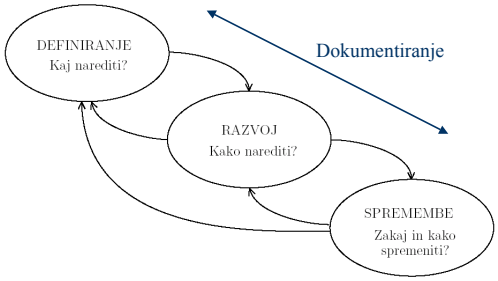
- Uvod v vodenje projektov
  - Programsko inženirstvo
  - Projektno delo
  - Osnovni razvojni modeli
  - Metode, ki sledijo načrtu
  - Agilne metode
- Dokumentiranje
  - Zagotavljanje varnosti
  - ISO 17799
  - **Razvojna dokumentacija**
  - **Uporabniška dokumentacija**

Programsko inženirstvo

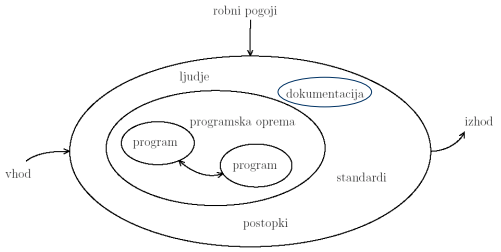


Nadzor v smislu doseganja načrtovane kvalitete, porabljenega časa in stroškov je slabši v primerjavi z drugimi tehniškimi področji!

Generičen razvojni cikel



Informacijsko načrtovanje



Obseg programskega inženirstva

- specifikacija programske opreme (analiza: kaj?)
- načrtovanje programske opreme (kako?)
- programerske tehnike in orodja
- validacija programske opreme
- vodenje projektov

⇒

Specifikacija programske opreme

- definicija zahtev (Software Requirements Definition)
- modeliranje sistema (System Modelling)
- specifikacija zahtev (Requirements Specification)
- strukturirane metode (neformalne: SSADM, SASD,...)
- formalne metode (stroga matematična podlaga: VDM, Z sheme,...)
- prototipiranje (Software Prototyping)
- ...

Načrtovanje programske opreme

- funkcijsko usmerjeno načrtovanje
- objektno usmerjeno načrtovanje
- načrtovanje sistemov v realnem času
- načrtovanje uporabniškega vmesnika
- ...

Programerske tehnike in orodja

- zanesljivost programske opreme
- ponovna uporaba programske opreme
- CASE (Computer Aided System Engineering) orodja
- okolja za razvoj programske opreme (Software Design Environment)
- ...

Validacija programske opreme

- verifikacija in validacija
- zanesljivost programske opreme
- varnost
- testiranje (ugotavljanje napak)
- orodja za testiranje
- statično preverjanje
- ...

Vodenje projektov

- izdelava planov
- tehnike mrežnega planiranja
- ocenjevanje stroškov in časa razvoja
- vzdrževanje programske opreme
- upravljanje s konfiguracijo (Configuration Management)
- **izdelava dokumentacije**
- zagotavljanje kakovosti
- ...

Projektno delo

Projekt je **enkratna,**
praviloma **zahtevna** in **kompleksna** skupina nalog,
ki mora biti končana v **določenem roku,**
doseči mora vnaprej določene in morebitne kasneje odkrite **cilje**
ter upoštevati **omejitve.**

Vrste projektov

- Deterministični (določnost) &
- Stohastični (empiričnost)

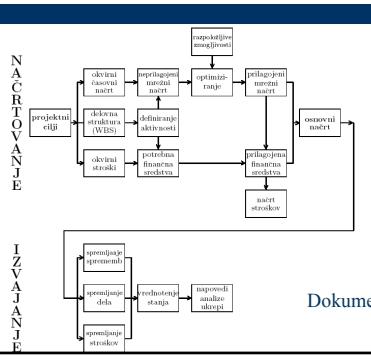
- Enkratni &
- Projektni procesi

Projektno : klasično upravljanje

- Upravljanje zajema načrtovanje, organiziranje, kadrovanje, vodenje in kontroliranje virov oziroma sredstev nekega podjetja, da bi dosegli finančne in druge zadane cilje tega podjetja.

- Klasično upravljanje: dolgoročnost in kontinuiranost dejavnosti
- Projektno upravljanje: specifično časovno obdobje in enkratni cilji

Delovne faze projekta



Skupina glavnega programerja

- Tako kot kirurg ali pilot tudi glavni programer sam opravlja vse bistvene naloge (analiza, načrtovanje, kodiranje), drugi člani ekipe pa mu pri tem asistirajo.
- Glavni asistent ga lahko kratkoročno nadomesti, tretji član skupine pa skrbi za administracijo in dokumentacijo.
- Skupini se lahko (občasno) pridružijo tudi drugi eksperti.
- Tehnična kompetentnost+voditeljske sposobnosti
- Pri velikih projektih v vlogi glavnega programerja nastopa skupina enakovrednih strokovnjakov, ki vodijo in nadzorujejo vse večje skupine programerjev.

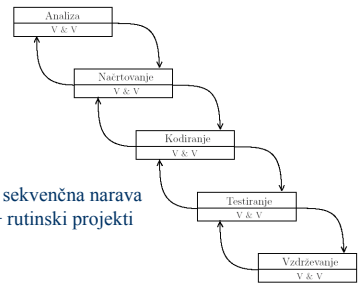
Ključne naloge projektne skupine

- podrobna in skrbna seznanitev z vsebino projektne naloge, z vsemi njenimi danostmi (standardi, glavni projekt, skupne informacijske osnove itd.), omejitvami in dokumentacijo ter drugimi informacijskimi viri, določenimi v odločbi;
- analiza naročnikovih zahtev, želja in pričakovanj kot izhodišče za določitev problemov in pričakovanih rešitev;
- izdelava vsebinske strukture in podrobnejšega načrta projekta z rokovnikom in imeni nosilcev posameznih aktivnosti ter s predračunom stroškov celotnega projekta;
- sestava okvirnega modela (podatkov in procesov) novega informacijskega sistema na podlagi popisa in analize obstoječih procesov, postopkov ter informacijskih in materialnih tokov (idejni projekt);
- izdelava projekta, njegova izvedba in prenos v prakso;
- seznanitev in usposobitev izvajalcev, uporabnikov in vzdrževalcev za izvajanje in vzdrževanje projekta;
- izročitev projekta v vzdrževanje z vso potrebno dokumentacijo;
- izdelava ocene o uresničitvi postavljenih ciljev, rokov in stroškov;
- usklajevanje dela z naročnikom ter seznanjanje le-tega s stanjem projekta in z uresničevanjem zastavljenih planov.

Osnovni razvojni modeli

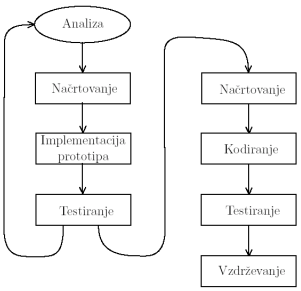
- Klasični razvojni cikel
- Razvoj z uporabo prototipov
- Razvoj s 4. generacijo programskih jezikov
- Postopni razvoj programske opreme
- Spiralni razvoj programske opreme
- Metode, ki sledijo načrtu
- Agilne metode

Klasični razvojni cikel



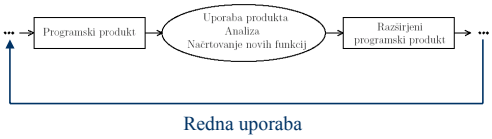
- sekvenčna narava
+ rutinski projekti

Razvoj z uporabo prototipov



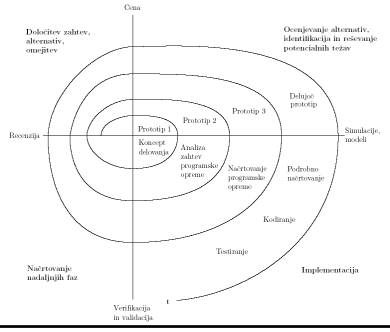
Kadar zahteve niso povsem jasno določene...

Postopni razvoj



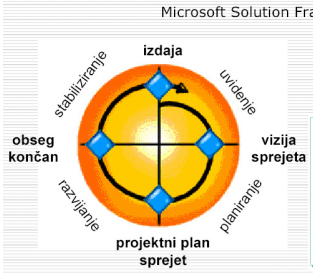
Redna uporaba

Spiralni razvoj



Hibrid

Primer iz prakse...



Metode, ki sledijo načrtu #1

- Primarni cilji: **napovedljivost**, **stabilnost**, **visoko jamstvo**
- Velikost: **velike skupine in projekti**
- Okolje: **stabilno**, **malo sprememb**, **osredotočenost na organiziranost in projekt**
- Naročnik: **interakcija po potrebi**, **osredotočenost na izvajanje pogodbe**
- Načrtovanje in nadzor: **dokumentirani načrti**, **numerična kontrola**
- Komunikacija: **eksplicitno dokumentirano znanje**

Metode, ki sledijo načrtu #2

- Tehnične zahteve: **formaliziran projekt**, **zmogljivost**, **vmesnik**, **kvaliteta**, **napoved evolucijskih zahtev**
- Razvoj: **obsežen načrt**, **dolgi inkrementi, preoblikovanje je privzeto kot draga naloga**
- Testiranje: **dokumentirani testni načrti in postopki**
- Razvijalci: **polovica** visoko usposobljenih na začetku, nato le **10%**, ostalo **programerji** z zmožnostjo sledenja metodi

Metode, ki sledijo načrtu #3

- Kultura dela: udobje in moč skozi ogrodje **načrtov** in **reda**

Agilne metode #1

- Primarni cilji: **hitro vidni rezultati**, **odzivnost na spremembe**
- Velikost: **manjše skupine in projekti**
- Okolje: **turbulentno**, **veliko sprememb**, **osredotočenost na projekt**
- Naročnik: **kolociran**, **osredotočenost na prioritetne inkremente**
- Načrtovanje in nadzor: **interni načrti**, **kvalitativna kontrola**
- Komunikacija: **zavedanje medosebnega (skupnega) znanja**

Agilne metode #2

- Tehnične zahteve: **prioritetizirane neformalne zgodbe in testi, predvidevanje novih sprememb**
- Razvoj: **enostaven načrt**, **kratki inkrementi**, **preoblikovanje je privzeto kot poceni naloga**
- Testiranje: **avtomatski testi definirajo zahteve**
- Razvijalci: **30%** visoko usposobljenih, ostalo **dobri programerji** z zmožnostjo sledenja metodi

Agilne metode #3

- Kultura dela: udobje in moč skozi ponujeno **svobodo** pri delu

Praksa

- Problem (neuspeh) **metod, ki sledijo načrtu**, je velika poraba časa in nefleksibilnost (spremembe) oziroma njena povezava s ceno
- V praksi se te metode redko dosledno uporabljajo (analiza=paraliza)
- Poudarek na izdelkih in procesih, bistveno manj na ljudeh

Statistika

- 40% projektov propade ali pa so opuščeni
- 80-90% programske opreme ne ustreza začrtanim ciljem
- 80% je dostavljene po izteku roka in prekorači proračun
- 25% pravilno integrira poslovne in tehnične cilje
- **Le 10-20% ustreza kriterijem uspeha**

Manifest za agilen razvoj

- **Posamezniki in interakcije med njimi** pomembneje od procesov in orodij
- **Delujoči programi** pomembneje od podrobne dokumentacije
- **Sodelovanje naročnika in razvijalcev** pomembneje od pogajanja o pogodbi
- **Pripravljenost na spremembe** pomembneje od sledenja načrtu

http://www.agilemanifesto.org

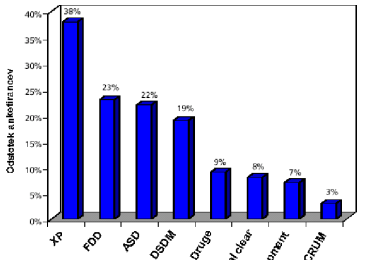
Povzetek

Metoda je agilna, ko je razvoj programske opreme:

- **Inkrementalen** – majhne izdaje, hitri razvojni cikli ⇒ hitre povratne informacije
- **Kooperativen** – naročnik in razvijalci tesno sodelujejo
- **Neposreden** – metoda je enostavna, torej lahko naučljiva, prilagodljiva in dobro opisana
- **Prilagodljiv** – lahko obvlada spremembe zahtev

Fleksibilnost & odzivnost

Agilne metode

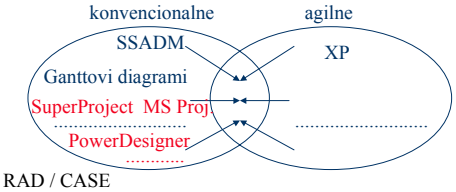


XP – eXtreme Programming
FDD – Feature Driven Development
ASD – Adaptive Software Development
DSDM – Dynamic Systems Development Method
...

dX ⇐ Rational Unified Process (RUP)
OSSD – Open Source Software Development

Metode razvoja – globalno

(SW-)CMM ISO 9000 ISO 17799



RAD / CASE

UML RationalRose

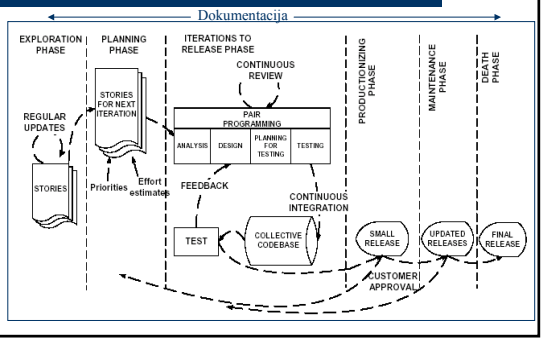
Opomba: metode oz. orodja lahko pokrivajo le del razvoja

XP – eXtreme Programming

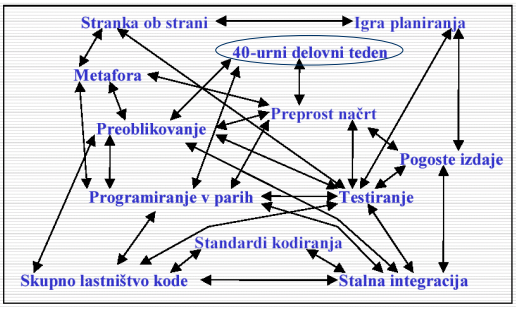
- Za majhne ali srednje velike skupine razvijalcev (med 2 in 20)
- Ob ohlapnih ali spreminjajočih se zahtevah
- Potreba in želja po hitri implementaciji poslovnih rešitev
- Najbolje deluje s sposobnim projektnim vodjem in skupino, ki je predana in izkušena in rešuje dobro definiran problem
- Ni za projekte, kjer je lahko ogroženo človeško življenje ali ključno premoženje

http://www.xprogramming.com

Življenski cikel XP procesa



Principi XP



Uporabniške zgodbe

- Uporabniški primeri, ki kratko opišejo zahteve za sistem z naravnim jezikom
- Berljive za naročnika in razvijalca
- Ni nujno da so popolne; obljuba za nadalnje pogovore
- Za vsako značilnost (feature) trije stavki – kratek opis, ki ga v osnovi pripravi naročnik
- Nadomestilo za zahtevnike oz. dokumente zahtev v konvencionalnih metodah
- Razlika je v podrobnostih; ko nastopi ustrezen trenutek, razvijalec in naročnik sedeta skupaj in razširita osnovni opis
- Naročnik določi prioritete
- Tipični XP projekt, ki traja od 6 do 12 mesecev, zahteva od 50 do 100 uporabniških zgodb!

Planiranje iteracij

- Naročnik razvrsti uporabniške zgodbe glede na poslovno vrednost in izbere nekaj zgodb z najvišjo vrednostjo; podrobna določitev prioritet
- Razvijalci ocenijo potreben čas na osnovi: prejšnjih iteracij, hitrih prototipov, izkušenj
- Iteracije niso mejniki v časovnem planu!
- 4. spremenljivke: stroški, čas, obseg, kakovost
- "XP-merji" planirajo
- Igra planiranja (up. zgodbe na karticah): cilji (plan izdaj različic), elementi igre (up. zgodbe), igralci (naročnik, razvijalci) in poteze (premikanje kartic)
- Srečanja ob začetkih iteracij (izbira uporabniških zgodb za to iteracijo)

Poudarek na skupinskem delu

- Lastnik projekta je celotna skupina in vsak član lahko sodeluje na kateremkoli delu
- Zagovorniki XP trdijo, da je par več kot 2× produktivnejši kot posameznik
- Programiranje v parih ("pair programming")

Programiranje v parih

- Ideja: dva programerja ki delata kot eden sta produktivnejša, kot če bi delala vsak zase, ker njun skupen izdelek vsebuje manj napak
- Vsa programska koda je razvita s pari razvijalcev; vsak par uporablja le en računalnik
- 2 vlogi: **voznik** in **navigator** – vlogi menjata
- Programiranje v paru ni enako sedenju pred zaslonom in opazovanju nekoga, ki tipka!!!
- Praksa??? Sočasno dokumentiranje?

http://www.pairprogramming.com

Testno voden razvoj

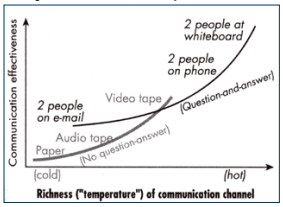
- **Najprej napišemo teste na osnovi zgodb, nato šele kodo, ki zgodbe realizira**
- Testiranje mora potekati **avtomatsko**
- Testi enote (Unit Tests – definirajo in izvedejo jih razvijalci)
- Funkcijski testi (Functional/Acceptance Tests – definirajo jih naročnik, izvedejo razvijalci)
- **Da napišemo teste, moramo razumeti zgodbe**
- Avtomatsko testiranje kot osnova za **preoblikovanje** (refactoring)

Preoblikovanje kode

- Preoblikovanje: spreminanje programske kode z namenom izboljšanja njene notranje strukture, brez sprememb zunanjega delovanja
- Preprečevanje entropije – slabšanje strukture programske kode
- Izvorna koda mora ostati preprosta in lahka za razumevanje
- **Najpreprostješa koda, ki uspešno opravi vse teste**
- **Koda je načrt in načrt je koda!**

Osebna komunikacija > Dokumentacija

- Ne pomeni nič drugega kot omejeno količino dokumentacije
- Način dokumentiranja: table z avtomatskim zapisovanjem, video zapisi



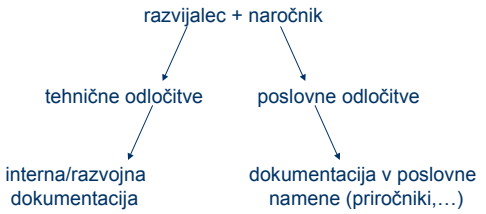
Delovno okolje

- Vsi v enem prostoru
- Že razporeditev v 2 nadstopji zmanjša komunikacijo
- Vsaj 17” ekran za programiranje v parih

Skupno lastništvo kode

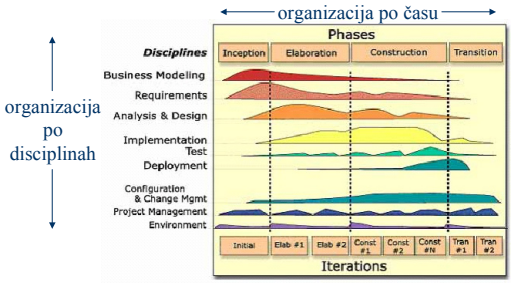
- Vsakdo lahko popravi katerikoli del kode
- Striktno upoštevanje dogovorjenih standardov kodiranja
- Rotiranje parov razvijalcev

Dokumentacija – vsakemu svoje



Odločitve o posamezni dokumentaciji so stvar prve ali druge skupine!

Rational Unified Process



Ali je RUP preobsežen?

- 9 disciplin
- Več kot 40 vlog (odgovornost, veščine,...)
- Stotine izdelkov (modeli, koda, dokumenti,...)
- Tisoče strani navodil (Kako izvesti neko aktivnost s pomočjo nekega orodja? – analiza=paraliza?)
- Potrebna prilagoditev
- V načrtu povežemo posameznike z vlogami, kar pomeni, da ni potrebno vsem poznati celotnega RUP-a.

Sledi...

Dokumentiranje!!!

Oris – ponovno, a podrobneje

- Zagotavljanje varnosti
  - Sistemska varnost
  - Zaupnost

  ⇒
- ISO 17799 – povezava z dokumentiranjem
- **Razvojna dokumentacija**
  - UML
  - Dokumentiranje kode
- **Uporabniška dokumentacija**

  Pisanje brošur, specifikacij, priročnikov,...

Sistemska varnost

- Varovanje lastnih, partnerjevih in naročnikovih podatkov
- Transakcijska neoporečnost
- Ocena tveganja preko varnostne revizije
- Upravljanje tveganja (Risk Management)
- IBM Tivoli (orodje, velika skupina izdelkov)

Zaupnost podatkov in informacij

- Moralna odgovornost!!!
- “need to know basis”

ISO 17799

- Standard varovanja informacij (Information Security Standard)
- Cilj: vpeljali učinkovit sistem za upravljanje informacijske varnosti (Information Security Management System)
- BS 7799: ISO/IEC 17799 + BS 7799-2:2002

BS 7799

Upoštevanje določil standarda BS 7799 nam omogoča, da bomo v podjetje vpeljali učinkovit sistem za upravljanje informacijske varnosti (ISMS), saj nam zagotavlja:

- ZAUPNOST – določena informacija je dostopna samo tistemu, ki mu je namenjena
- CELOVITOST – točnost in popolnost informacije ter metod obdelave le teh
- DOSTOPNOST – da imajo avtorizirani uporabniki dostop do informacije in s tem povezanih sredstev, kadarkoli je to potrebno

Kako zagotovimo varnost?

- Postopki načrtovanja, izvedbe, kontrole in popravila oziroma stalne izboljšave
- PDCA (Plan, Do, Check, Act)
- Glavni **dokumenti** pri vzpostavitvi standarda:
  - Upravljanje tveganja
  - Izjave o primernosti nadzorstev
  - Načrti izboljšav

  ⇐ Analiza tveganja predstavlja nosilni del standarda!

10 osnovnih poglavij/faz standarda

- *Business Continuity Management*
- *System Access Control*
- ***System Development and Maintenance***
- *Physical and Environmental Security*
- *Compliance*
- *Personnel Security*
- *Security Organization*
- *Computer & Operations Management*
- *Asset Classification and Control*
- *Security Policy*

Razvojna dokumentacija in UML

- RUP uporablja UML skozi celoten proces
- Preobsežnost RUPa?

- Prilagoditev
- Uporaba UML pri načrtovanju na najvišjem (npr. zasnova) in najnižjem (npr. podatkovne strukture) nivoju?

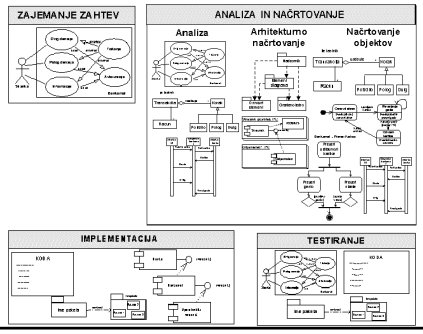
Unified Modeling Language

- Uporabniške zahteve (*diagram primerov uporabe*)
- Statična slika sistema (*razredni diagram*)

- Obnašanje programskega sistema (*diagram prehajanja stanj, diagram aktivnosti*)
- Interakcijo objektov (*diagram zaporedja, diagram sodelovanja*)

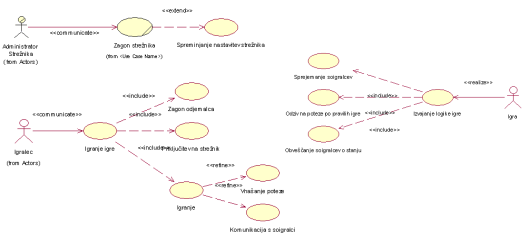
- Implementacijski konstrukti (*komponentni diagram, paketni diagram, diagram razvoja in dobave*)

Tipična uporaba posameznih diagramov

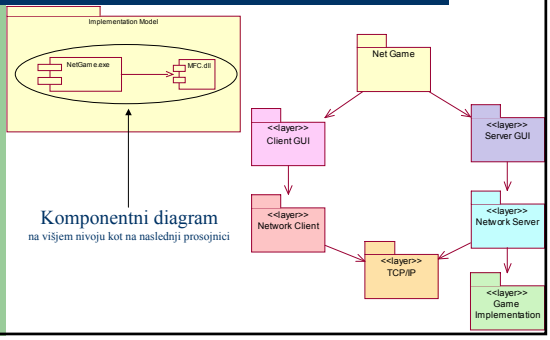


primeri posameznih diagramov ⇒

Diagram primerov uporabe

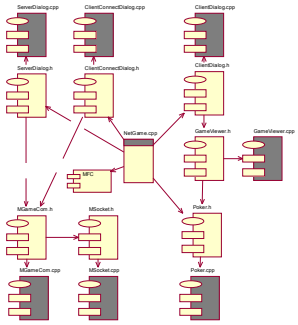


Paketni diagram



Komponentni diagram
na višjem nivoju kot na naslednji prosojnici

Komponentni diagram



Razredni diagram

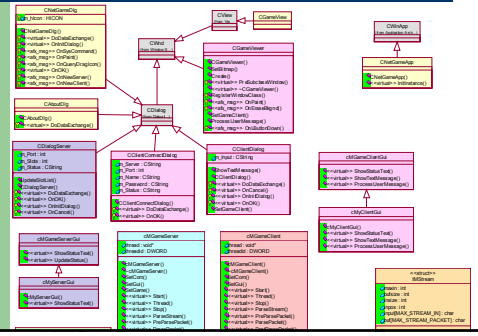


Diagram prehajanja stanj

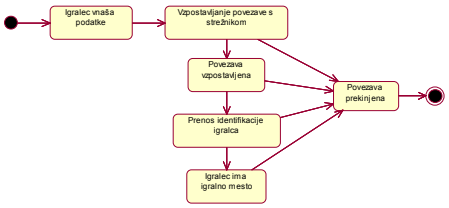


Diagram zaporedja

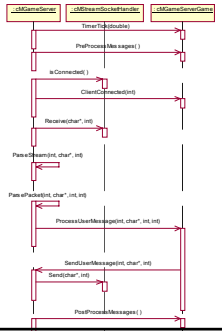


Diagram sodelovanja

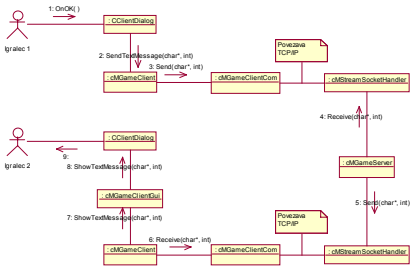


Diagram razvoja in dobave

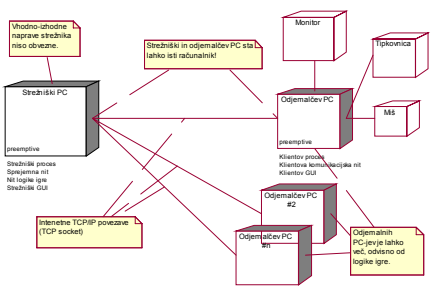
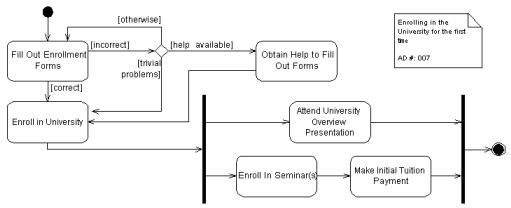


Diagram aktivnosti



Dokumentiranje kode

- Koda je dokumentacija!!! (XP)

- Struktura
- Preglednost; razdrobljenost na dele
- Berljivost kode
- Način poimenovanja spremenljivk,...
- Komentarji na vseh nivojih
- Izoliranost; združevanje sorodnih funkcij

Vsebina glave funkcije

- datum in ime avtorja, ki je kreiral funkcijo,
- datum, avtorja in namen modifikacij,
- opis delovanja funkcije,
- opis uporabljenih algoritmov,
- posebnosti pri izvedbi (če so) – še posebej tiste, ki lahko povzročijo probleme pri spreminjanju kode,
- seznam uporabljenih funkcij (samo funkcij, ki niso del standardnih knjižnic),
- opis vhodnih parametrov funkcije,
- opis izhodne vrednosti funkcije,
- uporabljene globalne spremenljivke (če se jih uporablja)

Primer glave funkcije

```
/*****************************************************************************
*                                                                            *
*          V p i s _ p a r a m e t r o v _ v _ E E P R O M                   *
*                                                                            *
* Modul: 1.09    Vpis parametrov regulatorja v EEPROM.                       *
*                                                                            *
*============================================================================*
* Verzija Datum    Opis                                          Avtor       *
*----------------------------------------------------------------------------*
*  1  1992-04-02  Zacetna verzija                                B. Premzel  *
*                                                                            *
*============================================================================*
*                                                                            *
* Sintaksa:                                                                  *
*    int Vpis_parametrov_v_EEPROM(struct eeprom_tab*);                       *
*                                                                            *
* Parametri: podatkovna struktura - tabela parametrov                        *
*            (podatkovna struktura je deklarirana v "vars.h")                *
*                                                                            *
* Izhodna vrednost: 1 (TRUE) - pravilno vpisano                              *
*                   0 (FALSE) - napaka pri vpisovanju                        *
*                                                                            *
* OPIS:                                                                      *
*  Program vpise v EEPROM vse parametre, ki so podani v tabeli, ki je        *
*  v datoteki "eeprom.h".                                                    *
*  Napako javlja, ce je parametrov prevec ali ce pride do napake pri         *
*  vpisovanju v EEPROM (spr. time ost pri poskusu vpisovanja v EEPROM -      *
*  napaka v EEPROM-u all manjkajoc EEPROM.                                   *
*                                                                            *
*  Podatki so dodatno zasciteni s kontrolno vsoto, ki se jo racuna ob        *
*  shranjevanju in preverja pri nalaganju. Kontrolno vsoto se vpise v        *
*  EEPROM tik za parametri.                                                  *
*                                                                            *
*****************************************************************************/
```


Vsebina glave datoteke

- ime datoteke in kratek opis vsebine,
- podatki o avtorju, datumih in vsebini modifikacij,
- seznam vseh funkcij, ki so v tej datoteki,
- seznam glavnih podatkovnih struktur oz. glavnih skupin podatkovnih struktur,
- navodilo za prevajanje in povezovanje, če se le-to izvaja na neobičajen način (posebne opcije prevajalnika, povezovalnika,...),
- posebnosti pri uporabi funkcij te datoteke (npr. "Pred uporabo funkcij, ki delajo s podatkovnimi strukturami, je potrebno klicati funkcijo nalozi_podatke().")

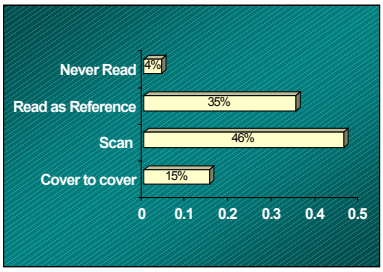
Primer glave datoteke

```
/*****************************************************************************
*                                                                            *
*                          E E P R O M . C                                   *
*                                                                            *
* Citaaje iz in pisaaje v EEPROM, ki je aa serijskem vodilu (I2C).           *
*                                                                            *
*============================================================================*
* Verzija Datum    Opis                                          Avtor       *
*----------------------------------------------------------------------------*
*  1  1992-04-10  Zacetna verzija                                B. Premzel  *
*  2  1992-12-03  Predelava za novo verzijo regulatorja          B. Premzel  *
*                                                                            *
*============================================================================*
*                                                                            *
* OPIS:                                                                      *
* Tukaj so zdruzene vse funkcije, ki shranjujejo informacije v EEPROM ali    *
* jih nalagajo iz njega:                                                     *
*                                                                            *
* void Brisanje_EEPROMa(int kaj);                                            *
*      Brisanje celotnega EEPROM-a (kaj = 1) ali samo shranjenih napak (0).  *
*                                                                            *
* void Zabelezi_cas_delovanja(void);                                         *
*      Pristejo cas delovanja regulatorja v vsoto, ki je v EEPROM-u.         *
*                                                                            *
* void Save_error(int vrsta_napake, unsigned int dodatna_inf_o_napaki);      *
*      Shrani informacije o napaki, ki je povzrocila izklop.                 *
*                                                                            *
* Funkcije, ki delajo direktno z aparaturno opremo, so v datoteki "I2c.a"    *
* in so realizirane v zbirnem jeziku (komunikacija z EEPROM-om).             *
*                                                                            *
*****************************************************************************/
```

Uporabniška dokumentacija

- navodila za namestitev
- konfiguracijski podatki
- uporabniški **priročnik**

Kako uporabniki berejo priročnike?



Razvojni cikel nastajanja publikacij

Koračni model:

1. Načrtovanje
2. Specifikacija vsebine
3. Pisanje
4. Produkcija
5. Ovrednotenje

Načrtovanje

- Zberi informacije
- Uredi informacije v osnutek
- Preveri usnutek (potrditev)

- Glavna naloga: načrt za **cel** dokument
- Torej, nikoli ne začni s pisanjem prvega odstavka!
- Določi osnovni namen publikacije (opis arhitekture, funkcionalnosti, aplikacije novega orodja,...)
- Kakšne informacije potrebuje bralec?
- Horizontalno načrtovanje

Specifikacija vsebine

- Zberi informacije
- Spiši specifikacijo vsebine
- Preveri specifikacijo (potrditev)

Vsebina specifikacije:

- Namen dokumenta
- Ciljno občinstvo
- Funkcija dokumenta
- Povezava z drugimi dokumenti
- Struktura dokumenta
- Povzetek vsebine
- Glavne odvisnosti in tveganja

Pisanje

- **Določi format publikacije**
- Spiši, uredi in ilustriraj
- Preveri osnutek (Edit Draft)
- Ustvari indeks

Produkcija

- Prevod in lokalizacija
- Pripravi končno različico:
  - Zadnji pregled (Final Edits)
  - Ustvari indeks in kazalo
- Sodelovanje s prodajalcem

Ovrednotenje

- Spremljanje odziva
- Arhiviranje elektronskega materiala
- Arhiviranje natisnjenega materiala

Osnovna načela

- Kdo je naše občinstvo?
  - Strokovnjaki: usmeri jih direktno na problem-rešitev oziroma na del dokumentacije
  - Začetniki: usmeri jih na lažje dele dokumentacije
  - Inženirji, prodajalci, stranke,...
- (1) Kaj želimo opisati & *(2) kaj pričakujejo bralci?*
  (1) Katere informacije naj predstavim v dokumentu?
  (2) Moram prebrati celo knjigo?
  (1) Kako naj organiziram dokument, da bo logično strukturiran?
  (2) Kako hitro lahko najdem informacijo, ki jo iščem?
  (1) Kako naj predstavim informacije brez preveliko teksta?
  (2) Razumem sporočilo? Je jasno podano?
  (1) Kako predstaviti informacije na zadovoljiv način?
  (2) Ali je povezava med poglavji očitna?

Ocene

- Nove strani: 2.5 strani/dan
- Spreminjanje: 3.5 strani/dan
- Nespremenjene strani: 20 strani/dan
- Online teme: 3.5 teme/dan
- Prvi spisan osnutek nastane po 60% vsega načrtovanega časa: 20% - raziskava , 40% pisanje

Predstavitev informacij

- Tekst čez celo stran je utrudljiv – ustrezna oprema dokumenta
- Ustvarjanje uravnoteženih strani
- Konsistentna oznaka rubrik (poglavij)
- Uporaba slik, tabel in grafov kjer je to možno

Stil

- Vsebina je kar sporočamo, stil pa kako jo sporočamo!
- Piši enostavno, neposredno in točno
- Stavki naj bodo kratki
- Izogibaj se humorja in žargona
- **Bodi konsistenten**
- Predvidevaj vprašanja bralcev
- Izogibaj se opredelitvi spola (...*njegov* nasvet...)
- Zapisano naj zveni naravno

Še nekaj napotkov za konec #1

- Razlika v naslovih na različnih stopnjah naj bo različna vsaj za faktor 1,4
- Poudarek (bold) je ekvivalent enemu nivoju više
- Izpostavi poglavja – npr. uporaba črt, krožnih segmentov
- Uporaba do 20% senčenja naslovov
- Uporaba različnih pisav za naslove in ostal tekst
- Ne uporabljaj le velikih črk v naslovih
- Na dveh najvišjih stopnjah začni na naslednji strani

Še nekaj napotkov za konec #2

- Naslov na najvišjem nivoju naj bo vedno na novi, desni, lihi strani
- Naslovi se uporabljajo za hitro iskanje po dokumentu – namen: označevanje stopenj in vsebine
- Izogibaj se okrajšavam
- Ton pisanja naj bo ustrežljiv, a ne ravnodušen; nekako uraden
- Akcije, rezultati in komentarji naj bodo ločeni
- Opiši postopek pred podajanjem detajlnih korakov
- Navodila naj bodo varna

Še nekaj napotkov za konec #3

- Uporabljaj slikovni material, kjer je to možno – slika lahko pove več kot 1000 besed, pa tudi človek si tako podano informacijo lažje zapomni
- Ko uporabljaš zaslonske slike, ne uporabljaj resničnih imen računalnikov, uporabniških imen,...
- Uporabljaj tabele za smiselno strukturiranje in predstavitev informacij

Še nekaj napotkov za konec #4

- Uporabljaj barve, saj: pospešujejo iskanje, učenje je lažje, pomagajo pri sprejemanju odločitev,...
- Ne uporabljaj več kot sedem barv
- Barve uporabljaj konsistentno
- Uporabljaj barvne asociacije iz narave (rdeča – kri – nevarnost)
- Uporabljene barve naj imajo različno svetlost (barvna slepota, čb tisk)

Natisnjena : online dokumentacija

- Prednosti
- Slabosti
- Kdaj uporabiti kateri pristop?

Najboljša dokumentacija je dobro premišljena kombinacija natisnjenega in online materiala, kjer vsak medij izkorišča svoje pozitivne lastnosti!

Natisnjena dokumentacija #1

Prednosti:

- Prenosljivost
- Večja ločljivost
- Lažje zapisovanje opomb, označevanje,...

Natisnjena dokumentacija #2

Slabosti:

- Največkrat nerodno in slabo formatirana
- Stroški produkcije so povezani s tiskom
- Nova različica zahteva zamenjavo
- Iskanje informacije zahteva fizično preiskovanje

Natisnjena dokumentacija #3

Kdaj uporabimo ta pristop?

- Uvajalni vodiči (Getting Started)
- Vodiči za odpravljanje napak (Troubleshooting)
- Material za hitri vpogled (Quick Reference)
- Materiali, ki jih rabimo, ko nismo ob računalniku
- Materiali za namestitev

Online dokumentacija #1

Prednosti:

- Dobro orodje za iskanje informacij
- Ni stroškov s tiskanjem
- Spremembe lahko enostavno integriramo

Online dokumentacija #2

Slabosti:

- Zahteva računalnik, načeloma pa jo lahko tudi natisnemo
- Težja za uporabo za netehnične uporabnike
- Slabša ločljivost
- Razvojni čas je daljši
- Index in kazalo nista dosegljiva, če jo natisnemo

Online dokumentacija #3

Kdaj uporabimo ta pristop?

- Ko potrebujemo informacijo in delamo za računalnikom
- Za proceduralne naloge
- Za online odpravljanje napak

Sprotno dokumentiranje?

- Pojavljajo se primeri, kjer sprotno dokumentiranje za uporabnika pospeši celoten razvoj!
- XP + sprotno dokumentiranje za uporabnika ⇔ mogoče celo kot alternativa programiranju v parih?

Osnovna literatura

- M. Heričko: *Sodobni procesi razvoja*, prosojnice, FERI, llll?.
- A. Jaklič: *Agilne metodologije razvoja programske opreme*, predavanje, FRI, 2003.
- I. Jakša Zupančič: *User Documentation*, predavanje, HERMES SoftLab, 2001.
- M. Peternel: *Razvoj sodobne internetne računalniške igre*, seminarska naloga na podiplomskem študiju FRI, 2003.
- J. Pogorelc, M. Terbuc: *Specificiranje, dokumentiranje in izdelava programske opreme s programskim jezikom C*, zapiski predavanj, FERI, 2000.
- F. Solina: *Projektno vodenje razvoja programske opreme*, založba FE/FRI, 1997.
- L. Williams, A. Cockburn (gostujoča urednika): Agile Methods, *IEEE Computer*, junij 2003.
- Internet!