Načrtovanje

- Arhitektura sistema
- Razdelitev analitičnega modela v podsisteme
- Načrt uporabniškega vmesnika (identifikacija možnih akcij glede na uporabniške scenarije, modela obnašanja; definicija elementov uporabniškega vmesnika in kontrolne mehanizme)
- Načrt komponent (definicija algoritmov z relativno majhnim nivojem abstrakcije; pregled vmesnikov, definicija podatkovnih struktur na nivoju komponente)
- Razvoj načrta inštalacije

Arhitektura sistema

- Definicija relacij med glavnimi strukturnimi elementi aplikacije
- Je predstavitev, ki omogoča inženirjem:
  - analizo učinkovitosti načrta
  - analizo arhitekturnih alternativ
  - zmanjšati tveganja
- Olajša komunikacijo med naročniki in razvijalci
- Izpostavi začetne odločitve o načrtovanju
- Predstavlja relativno enostaven model strukture sistema

Arhitektura sistema

- Arhitekturni stil:
  - Podatkovno orientiran
  - Na osnovi pretoka informacij
  - Hierarhični
  - Nivojski
- Vzorci:
  - Paralelizem
  - Obstojnost podatkov
  - Porazdeljenost
- Organizacija:
  - Kontrola
  - Prenos podatkov

Načrt uporabniškega vmesnika

- Tri osnovna pravila:
  - uporabnik naj ima kontrolo nad vmesnikom
  - uporabnik naj si mora čimmanj zapomniti
  - vmesnik naj bo konsistenten
- Faze:
  - analiza uporabnikov, funkcionalnosti in okolja
  - načrt vmesnika
  - izdelava
  - evaluacija

Načrt komponent

- Komponenta je modularen in nadomestljiv del celotnega sistema
- V OO notaciji komponenta predstavlja skupek enega ali več razredov
- V klasični notaciji predstavlja del programa (procesno logiko, interne podatkovne strukture, vmesnike in vhodne podatke)
- V spletnih aplikacijah komponenta predstavlja posamezno stran aplikacije
- Komponenta naj bo odprta za razširitve, zaprta za modifikacije!
- Posamezen razred naj bo zamenljiv s svojim baznim razredom
- Komponenta naj bo odvisna od abstraktnih definicij in ne od konkretnih
- Več prillagodljivih vmesnikov je bolj kot en sam splošen
- Razredi, ki se spreminjajo skupaj, sodijo v isto komponento
- Razredi, ki se ne uporabljajo skupaj, ne sodijo v isto komponento

Načrt komponent

- Konsistentno poimenovanje komponent, vmesniki za komunikacijo med komponentami, odvisnost in dedovanje
- Pretirana povezanost komponent (direktno spreminjanje atributov razreda druge komponente, uporaba globalnih spremenljivk, izvajanje operacij druge komponente, itd)

Postopek

- Identifikacija vseh razredov funkcionalne domene
- Identifikacija vseh razredov infrastrukturne domene
- Podrobna obravnava vseh razredov, ki niso že narejeni:
  - Način komunikacije z ostalimi razredi/komponentami
  - Definicija atributov, njihovih tipov in možnih in začetnih vrednosti
  - Detajlni opis vsake operacije (psevdokoda ali aktivnostni UML diagram)
- Opis trajnih virov podatkov in razredov za upravljanje z njimi (tabele v podatkovni bazi, datoteke, ipd)
- Detajlni opis obnašanja razredov ali komponente (UML diagram prehajanja stanj ali v posebni tekstovni obliki)
- Evaluacija načrta in iskanje alternativ

OCL in psevdokoda

- Definirata formalno gramatiko za zapise v nedvoumni obliki
- OCL se uporablja za zapis pogojev in predpogojev za različne elemente načrta (razrede, dogodke, sporočila, vmesnike, ipd)
- Psevdokoda se uporablja za zapis operacij