TEHNIČKO VELEUČILIŠTE U ZAGREBU
POLYTECHNICUM ZAGRABIENSE

Objektno orijentirani razvoj programa

ISVU: 130938/19970

Dr. sc. Danko Ivošević, dipl. ing.
Predavač

Ak. god. 2021./22.
Ljetni semestar

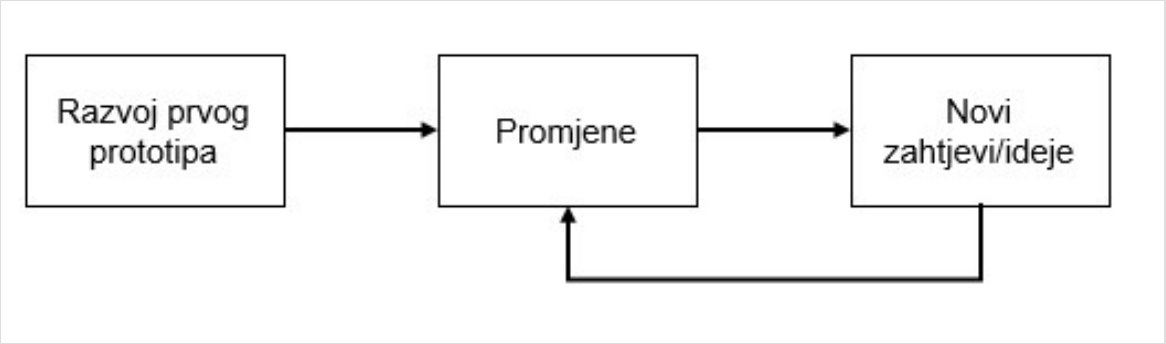
Procesi i modeli procesa programskog inženjerstva

Objektno orijentirani razvoj programa

Modeli procesa programskog inženjerstva

- dobre prakse pristupa izradi programske potpore
- iskustva razvojnih timova kao skup pozitivnih zaključaka o važnosti i redoslijedu izvođenja projektnih aktivnosti

ad-hoc ili **oportunistički pristup**



ad-hoc ili **oportunistički pristup**

- Nema razrade zahtjeva i oblikovanja prije same implementacije.
- Dolazi do brže istrošenosti.
- S obzirom na neplanski razvoj, cilj nikad nije dovoljno jasan.
- Proces ispitivanja ni bilo koji oblik potvrde očekivane kvalitete proizvoda nisu unaprijed ni izričito izdvojeni
- trošak razvoja i održavanja je vrlo visok.

Generički modeli programskog inženjerstva

- **vodopadni model** (engl. *waterfall model*)
  - temeljne aktivnosti procesa izrade programske potpore smatraju se nezavisnim fazama razvoja
- **evolucijski model** (engl. *evolutionary model*)
  - sustav se razvija kroz niz inačica ili inkremenata od kod kojih svaki sljedeći inkrement dodaje neku novu funkcionalnost u onu na koju se nastavlja
- **komponentni model** (engl. *component-based model*)
  - motivacija je u pretpostavci ponovne iskoristivosti postojećih komponenata.

Zasebni modeli programskog inženjerstva

- **Modelno-usmjereni razvoj**
  - temeljen na oblikovanju uporabom modela (engl. *model-based design*, *model-based software engineering*)
  - → **unificirani proces** (engl. *Unified Process, UP*)
- **agilni razvoj** (engl. *agile development*)

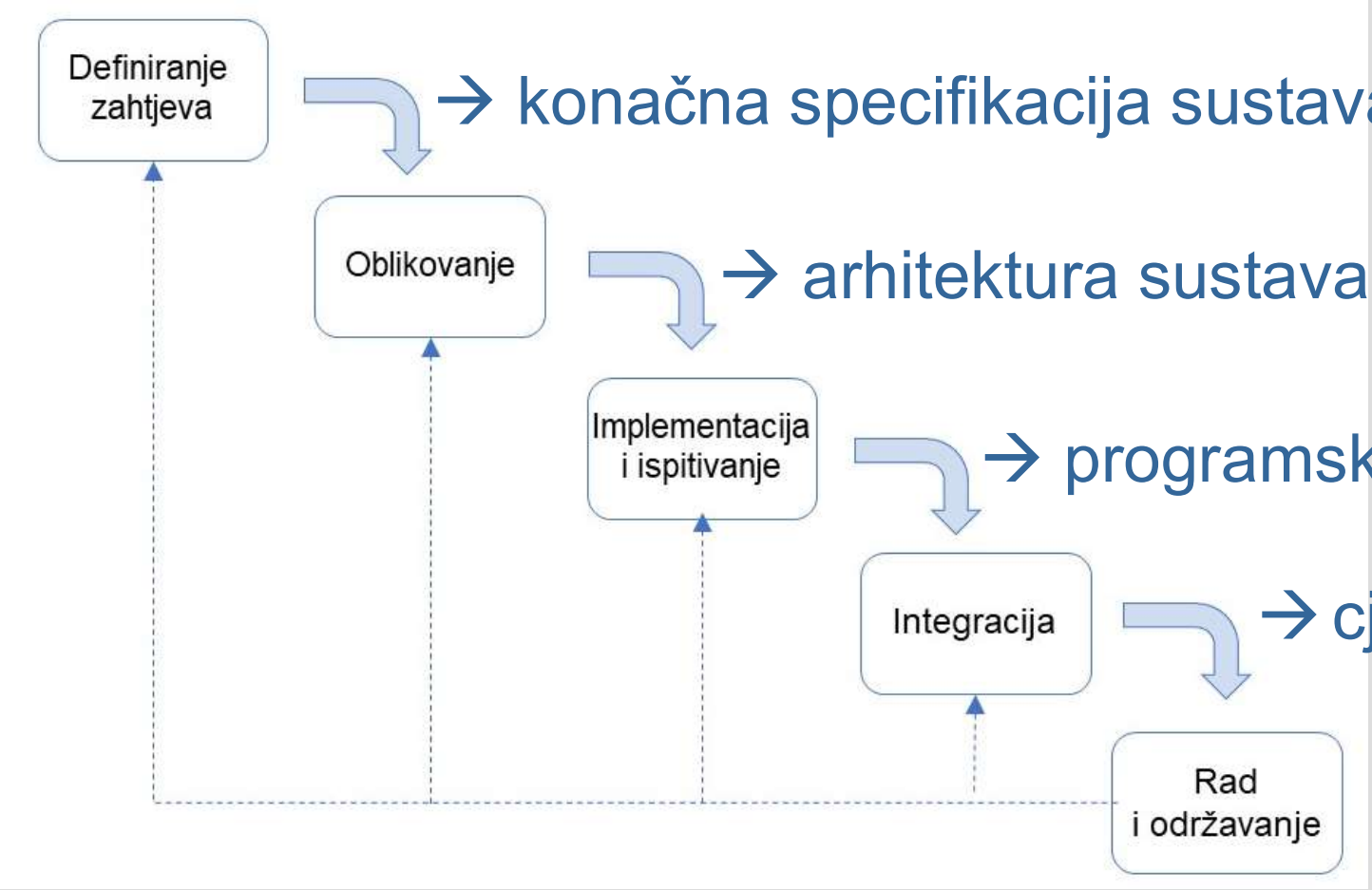
Procesi i modeli procesa programskog inženjerstva

VODOPADNI MODEL

Vodopadni model programskog inženjerstva

- procesne faze su jasno odvojene i u načelu nezavisne:
  - izlučivanja zahtjeva
  - oblikovanja
  - implementacije
  - integracije
  - održavanja sustava.
- dobro definiran plan rada za svaku od njih
- Sljedeća faza započinje tek kada je prethodna faza završena.

Vodopadni model programskog inženjerstva



Vodopadni model programskog inženjerstva

- strogost modela
- nezavisnost pojedinih faza
- ako su razumljivi zahtjevi s malom vjerojatnošću promjene
- razvoj na nekoliko odvojenih mjesta
- ako postoji formalna specifikacija sustava

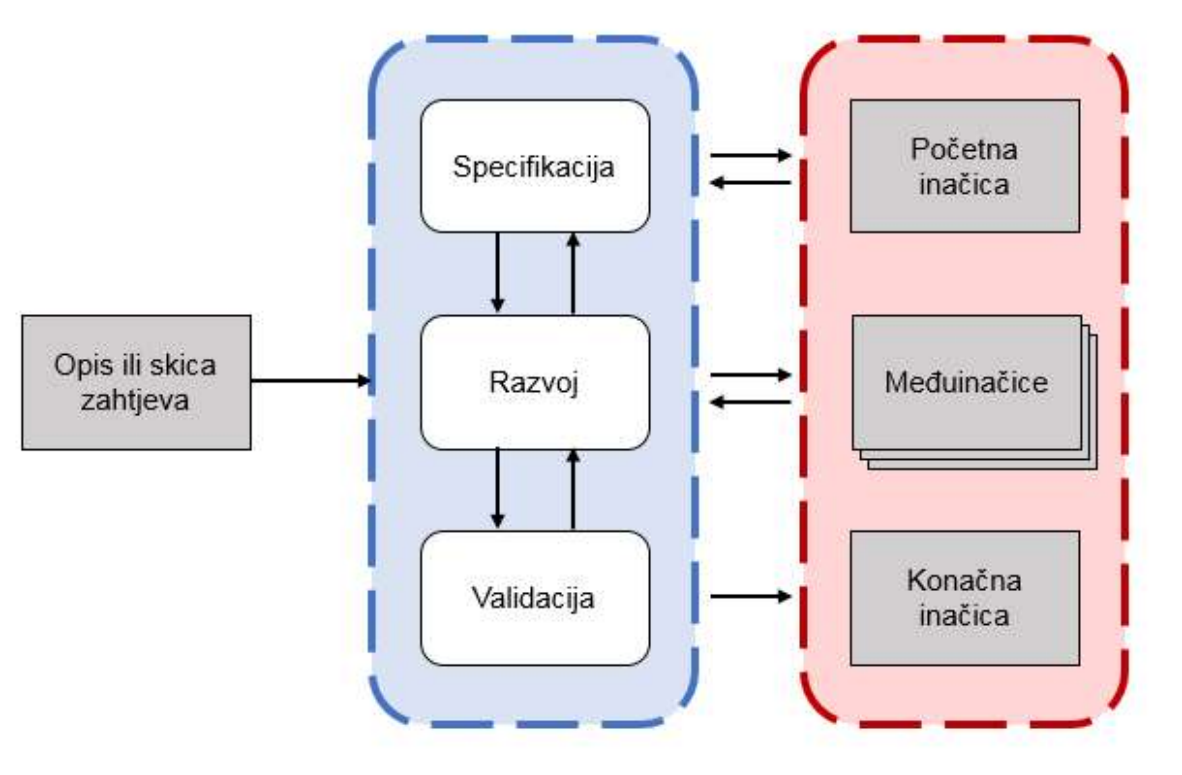
Procesi i modeli procesa programskog inženjerstva

EVOLUCIJSKI MODEL

Evolucijski model

- ispreplitanje faza:

  (ili projektnih aktivnosti?)



Evolucijski model

- Dva tipa:
  - istraživački razvoj i oblikovanje (engl. *exploratory development*)
  - metoda odbacivanja prototipa (engl. *throw-away prototyping*)
- „Slabije” definirani zahtjevi:
  - razvoj započinje u dijelovima sustava za koje su zahtjevi dobro definirani
  - izrada prototipova sustava koji se ne koriste u konačnici

Evolucijski model

- Prednosti:
  - brži nego vodopadni model
  - osnovica agilnog pristupa
    - brzi razvoj inkremenata
  - inkrementalni razvoj specifikacije (*)

Evolucijski model

- Nedostaci:
  - inkrementalni razvoj specifikacije (*)
  - proces razvoja i oblikovanja nije jasno vidljiv
  - zbog čestih izmjena narušeni:
    - struktura sustava
    - kvantifikacija napretka
    - dokumentiranje izdanja/inačica

Procesi i modeli procesa programskog inženjerstva

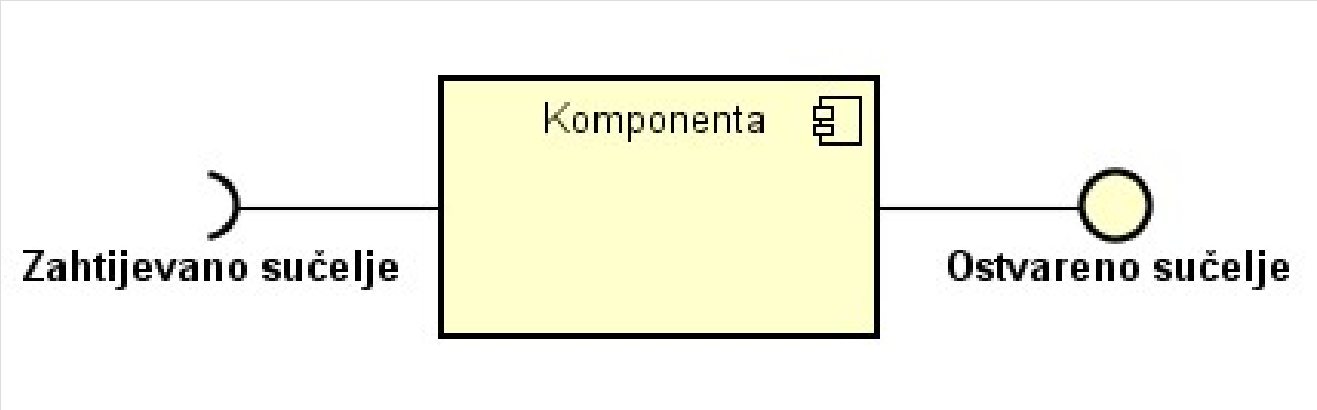
KOMPONENTNO-USMJERENI MODEL

Komponentno-usmjereni model

- Komponenta je nezavisna jedinica programske potpore s određenom funkcionalnosti koja se može kombinirati s drugim nezavisnim jedinicama u svrhu izrade cjelovitog sustava programske potpore.
- Komponenta kao „crna kutija”:
  - lokacija i tehnologija ostvarenja nebitni
  - identifikacija sučeljem

Komponentno-usmjereni model

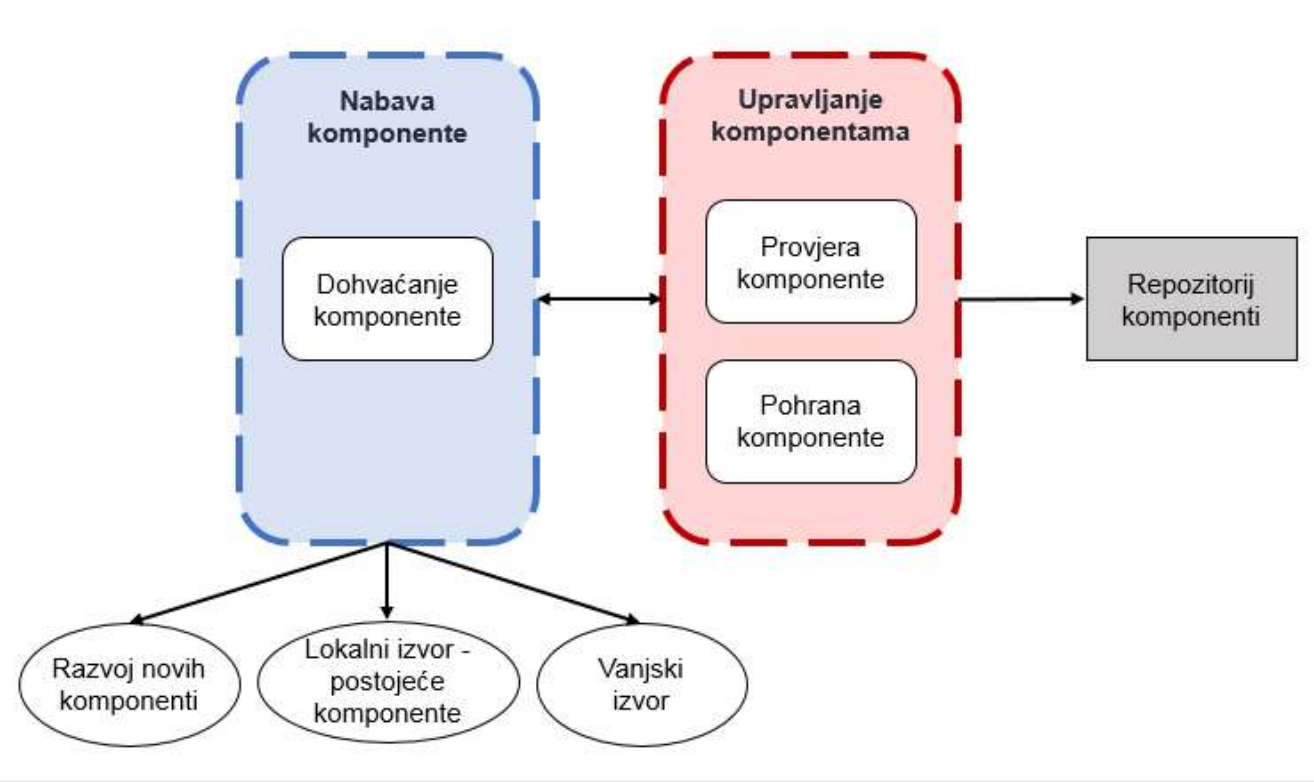
- Dva su temeljna tipa sučelja komponente:
  - **ponuđeno** ili izvezeno sučelje (engl. *exported interface*)
  - **zahtijevano** ili uvezeno sučelje (engl. *imported interface*)



Komponentno-usmjereni razvoj

- Dva različita stajališta:
  - razvoj za ponovnu iskoristivost (engl. *development for reuse*)
  - razvoj s korištenjem postojećih komponenti (engl. *development with reuse*)

Komponentno-usmjereni model



Komponentno-usmjereni model

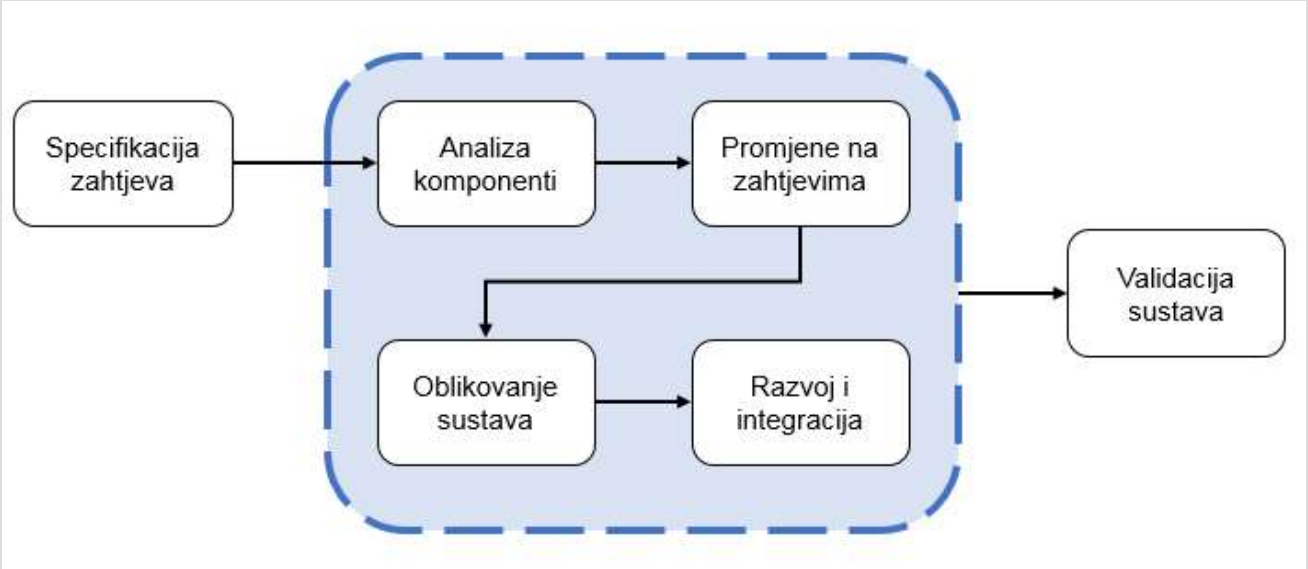
- Zadana norma komponente:
  - definicija sučelja
  - uputa za korištenje
  - uputa za isporuku

Komponentno-usmjereni model

- ponovno iskorištavanje postojećeg kôda se događa često i neformalno
- razlučivosti jedinica programske potpore (u praksi):
  - objekti
  - komponente aplikacije
  - gotova aplikacija

Komponentno-usmjereni model

- Razvoj temeljen na ponovnoj iskoristivosti komponenti:



Prednosti komponentno-usmjerenog modela

- povećana je pouzdanost komponente
- olakšana je, bolja i jasnija procjena troška
- specijalizirani razvoj: bolje poklapanje sa zadanim normama izrade komponente
- značajno ubrzanje u razvoju sustava

Mane komponentno-usmjerenog modela

- nedostupan izvorni programski kôd, troškovi održavanja komponente rastu ako postane nekompatibilna s promjenama u sustavu
- korištenje alata programske potpore koji ne predviđaju ili ne podržavaju koncept ponovne uporabnosti komponenti
- visoki troškovi održavanja knjižnice komponenata mogu biti visoki
- napor pronalaženja, razumijevanja i prilagodbe komponenata

Procesi i modeli procesa programskog inženjerstva

UNIFICIRANI PROCES (UP)

Unificirani proces

- Unificirani proces (engl. *Unified Process*, UP) je metodologija razvoja programske potpore koja se temelji na oblikovanju pomoću modela (engl. *Model Based Design*, MBD), odnosno iterativnom razvoju, obrascima uporabe i usmjerenjem na arhitekturu sustava.

Unificirani proces

- Faze životnog ciklusa:
  - početak (engl. *inception*)
  - elaboracija (engl. *elaboration*)
  - izgradnja (engl. *construction*)
  - prijenos proizvoda korisnicima (engl. *transition*)

Unificirani proces

- Faze životnog ciklusa:
  - početak (engl. *inception*)

    ➔ vizija ili ciljevi životnog ciklusa (engl. *lifecycle objectives*)
  - elaboracija (engl. *elaboration*)

    ➔ temeljna arhitektura (engl. *lifecycle arhitecture*)
  - izgradnja (engl. *construction*)

    ➔ početna sposobnost (engl. *initial operational capability*)
  - prijenos proizvoda korisnicima (engl. *transition*)

    ➔ izdavanje izvršne inačice programa ili proizvoda (engl. *release*)

Procesi i modeli procesa programskog inženjerstva

AGILNI PRISTUP RAZVOJU PROGRAMSKE POTPORE

Agilni pristup

- Agilni pristup razvoju programske potpore podrazumijeva skupinu metoda za razvoj programske potpore kojima je zajednički iterativni razvoj uz male inkremente i brz odziv na korisničke zahtjeve. Ovaj model razvoja programske potpore koristi se za razvoj manjih i srednjih projekata u stalnoj interakciji s klijentima putem stalnog predočavanja novih poboljšanja, uz relativno slabo dokumentiranje.

Manifesto for Agile Software Development

- *"We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:*

Manifesto for Agile Software Development

- ***Individuals and interactions*** *over processes and tools*
- ***Working software*** *over comprehensive documentation*
- ***Customer collaboration*** *over contract negotiation*
- ***Responding to change*** *over following a plan*
- *That is, while there is value in the items on the right, we value the items on the left more.“*

12 temeljnih načela agilnog razvoja

- Najvažnije nam je zadovoljstvo naručitelja, koje postižemo ranom i neprekinutom isporukom programskog proizvoda koji donosi vrijednost.
- Spremno prihvaćamo promjene zahtjeva, čak i u kasnoj fazi razvoja. Agilni procesi koriste promjene da naručitelju stvore kompetitivnu prednost.
- Često isporučujemo upotrebljiv programski proizvod, u razmacima od nekoliko tjedana do nekoliko mjeseci, nastojeći da razmak bude čim kraći.
- Poslovni ljudi i razvojni inženjeri moraju svakodnevno zajedno raditi, tijekom cjelokupnog trajanja projekta.

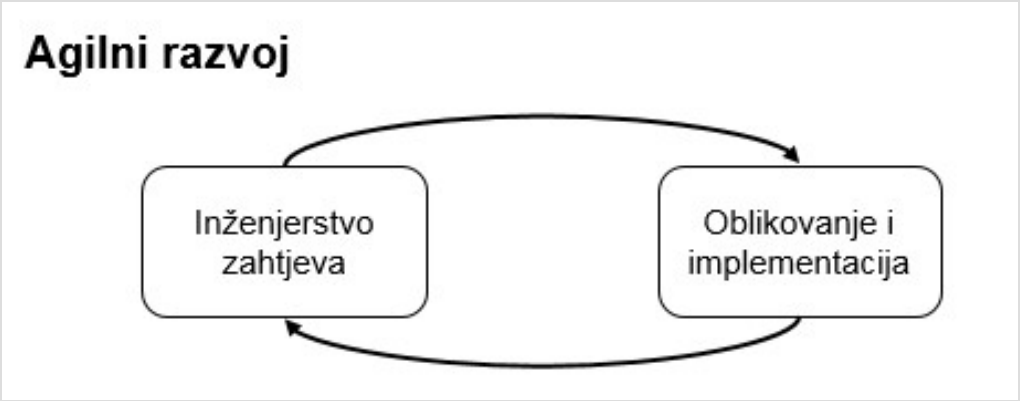
12 temeljnih načela agilnog razvoja

- Projekte ostvarujemo oslanjajući se na motivirane pojedince. Pružamo im okruženje i podršku koja im je potrebna, i prepuštamo im posao s povjerenjem.
- Razgovor uživo je najučinkovitiji način prijenosa informacija razvojnom timu i unutar tima.
- Upotrebljiv programski proizvod je osnovno mjerilo napretka.
- Agilni procesi potiču i podržavaju održivi razvoj. Pokrovitelji, razvojni inženjeri i korisnici trebali bi moći neograničeno dugo zadržati jednak tempo rada.
- Neprekinuti naglasak na tehničkoj izvrsnosti i dobrom oblikovanju pospješuju agilnost.

12 temeljnih načela agilnog razvoja

- Jednostavnost – vještina povećanja količine posla koji ne treba raditi – je od suštinske važnosti.
- Najbolje arhitekture, projektne zahtjeve i oblikovanje programske potpore stvaraju samo-organizirajući timovi.
- Tim u redovitim razmacima razmatra načine kako da postane učinkovitiji i zatim usklađuje i prilagođava svoje ponašanje u tom smjeru.

Model procesa agilnog razvoja



Metode/modeli agilnog razvoja

- Ekstremno programiranje (XP)
- Scrum
- “Čisti” razvoj (engl. *Lean development*)
- Kanban i Scrumban
- Disciplinirana agilna isporuka (DAD)
- Scrum na velikoj skali (LeSS)
- Adaptivni razvoj programske potpore (ASD)
- Crystal clear i ostale metode *crystal*
- Metoda dinamičnog razvoja sustava (DSDM)
- Razvoj vođen karakteristikama (FDD)
- Agile Unified Process (AUP)

Zaključno o agilnom razvoju

- naručitelj/korisnik spreman usko surađivati s razvojnim timom
- visok stupanj povezanosti članova razvojnog tima

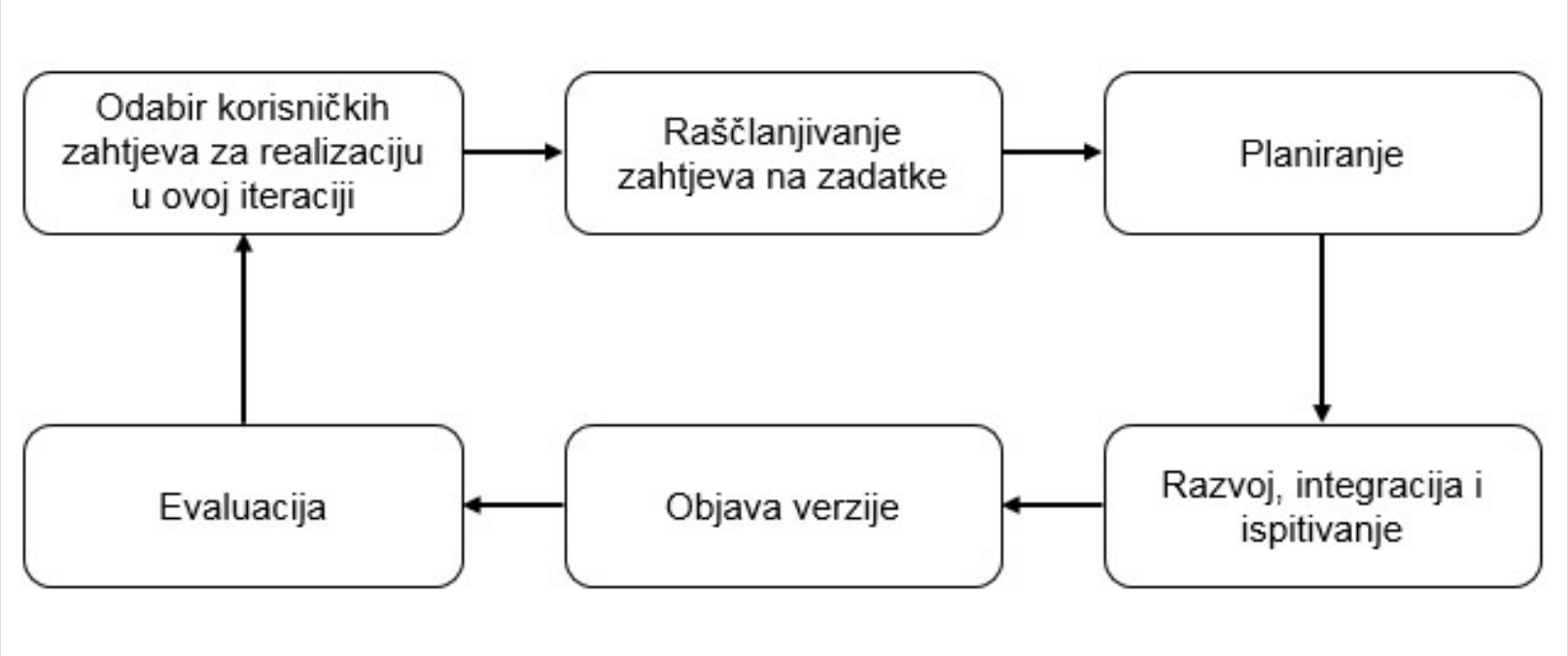
Za i protiv agilnog razvoja

- razvoj „velikih“ programskih sustava koji zahtijevaju suradnju između više razvojnih timova ili pak razvoj programske potpore s posebnim zahtjevima (sigurnost, pouzdanost, odziv u stvarnom vremenu)
- razvoj programske potpore koja će se koristiti kroz dugi vremenski period: manjak formalne i opsežne dokumentacije

Ekstremno programiranje (XP)

- jedina mjera napretka je funkcionalni programski proizvod
- razvoju, oblikovanju i brzoj isporuci funkcionalnih dijelova, uz stalno jedinično ispitivanje i često integracijsko ispitivanje
- stalno poboljšavanje koda i sudjelovanje korisnika/naručitelja u razvojnom timu
- programiranje u paru (*engl. pairwise programming*)

Ekstremno programiranje (XP)



Čisti razvoj programske potpore

- (engl. *Lean development*, *Lean*)
- cilj razviti u što kraćem vremenu sustav koji će maksimalno zadovoljiti korisnike uz minimalni nepotrebni angažman

Čisti razvoj programske potpore

- Principi oblikovanja:
  - eliminacija svega suvišnoga
  - naglašeno učenje
  - odlaganje donošenja odluke
  - brza isporuka
  - uvažavanje tima
  - ugradnja cjelovitosti u sustav
  - optimizacija cjeline

Scrum

- radni okvir agilnog razvoja programske potpore koji je strukturiran tako da podrži razvoj složenih proizvoda
- znanje dolazi iz iskustva i odlučivanja temeljenog na poznatome
- Timovi:
  - uloge, događaji, artefakti, pravila

Uloge u Scrumu

- vlasnik proizvoda (engl. *product owner*)
- Scrum vođa (engl. *Scrum master*)
- razvojni tim (engl. *development team member*)

Događaji u Scrumu

- (engl. *Scrum events*), osnovna jedinica razvoja **sprint:**
  - sastanak planiranja sprinta (engl. *sprint planning*)
  - sprint i dnevni sastanak Scrum tima (ili dnevni Scrum, engl. *daily Scrum, stand-up*)
  - revizija (ili recenzija) sprinta (engl. *sprint review*)
  - restrospektiva sprinta (engl. *sprint retrospective*).

Artefakti u Scrumu

- projektni dnevnik zaostataka (engl. *product backlog*)
- sprint dnevnik (engl. *sprint backlog*) je skup stavaka odabrane za Sprint
- inkrement

- → završenost se određuje nekim *pravilima* Scruma dogovorenima unutar svake tvrtke

REFERENCE I LITERATURA

- A. Jović, N. Frid, D. Ivošević: Procesi programskog inženjerstva, 3. izdanje, Sveučilište u Zagrebu, FER ZEMRIS, 2019.
- I. Sommerville, Software engineering, 10th ed., Pearson, 2016.
- G. Overgaard, B. Selic, C. Bock, OMG, 2000.
- S. Tockey: How to Engineer Software, Wiley-IEEE Computer Society Press, 2019.

Hvala na pažnji!

Pitanja?