Beogradska akademije poslovnih i umetničkih strukovnih studija

Projektovanje informacionih sistema

dr Rade Matić

Model

Model predstavlja subjektivnu, apstraktnu (uprošćenu) sliku sistema, opisujući elemente tog sistema i njegove veze.

Model sistema je uvek subjektivna slika sistema sagledana iz ugla gledanja posmatrača ili subjektivna slika objektivne stvarnosti.

Direktno postoji samo sistem, a svaki naš prikaz sistema je model.

Sistem se opisuje pomoću modela, odnosno, preko modela. Najprostiji primer modela je geografska karta sveta.

Model

Postoje dva **aspekta modela**: statički i dinamički aspekti modela, ali i **ograničenja** modela.

Statički aspekti modela se fokusiraju na njegovu strukturu (model podataka).

Dinamički aspekti modela se fokusiraju na njegovo ponašanje (model procesa).

Ograničenja opisuju ograničenja vezana za strukturu (npr. vrednosti ili kardinalnost atributa) i dinamičko ponašanje.

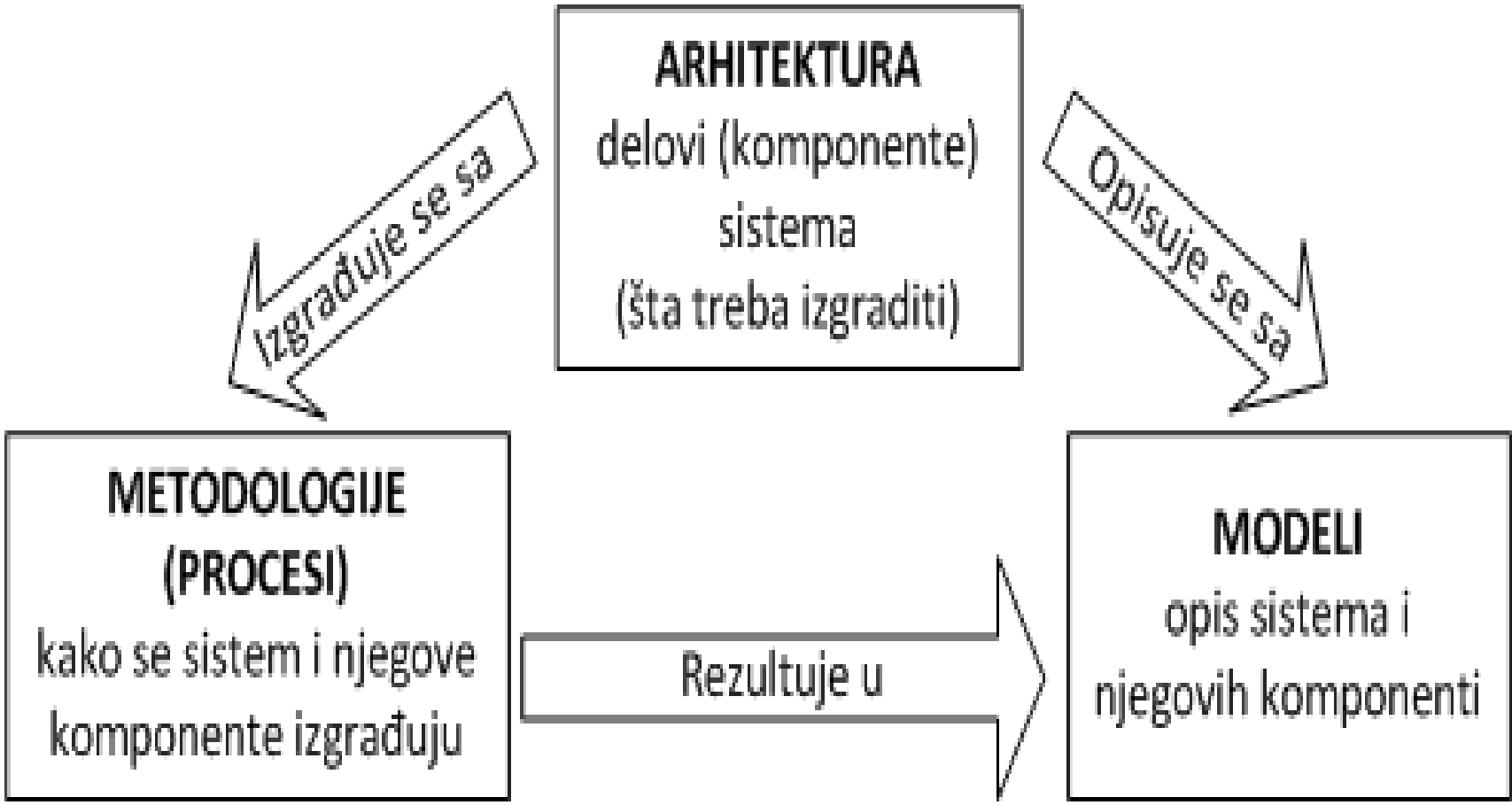
Modelovanje i metodologija

Modelovanje sistema predstavlja **postupak** kreiranja **modela sistema**.

Postupak kreiranja modela sistema se **definiše i opisuje metodom modelovanja**.

Osnovni elementi informacionog sistema

Svaku metodologiju određuju tri koncepta, tj. osnovna elementa IS: **procesi, modeli, arhitektura.**



Odnos arhitekture, procesa i modela

Osnovni elementi informacionog sistema

1. dekompozicija složenog sistema celokupan sistem se podeli na manje, lakše savladive celine tj. na međusobno povezane podsisteme (delove, komponente) – čime se definiše **arhitektura** sistema. Najzastupljenije metode za dekompoziciju sistema su funkcionalna i objektna dekompozicija.

2. a zatim se i sam **proces** razvoja IS (metodologija razvoja) podeli na faze (**model životnog ciklusa**), čime se daje skup neophodnih aktivnosti i njihov redosled i način izvođenja, a koje se javljaju tokom celokupnog života nekog IS.

3. **Modeli** sistema predstavljaju presek ova dva koncepta, jer se sistem, odnosno arhitektura sistema opisuje modelima, a modeli se izgrađuju u procesu razvoja i oni su ti koji neki IS opisuju sa različitih tačaka posmatranja i nivoa apstrakcije.

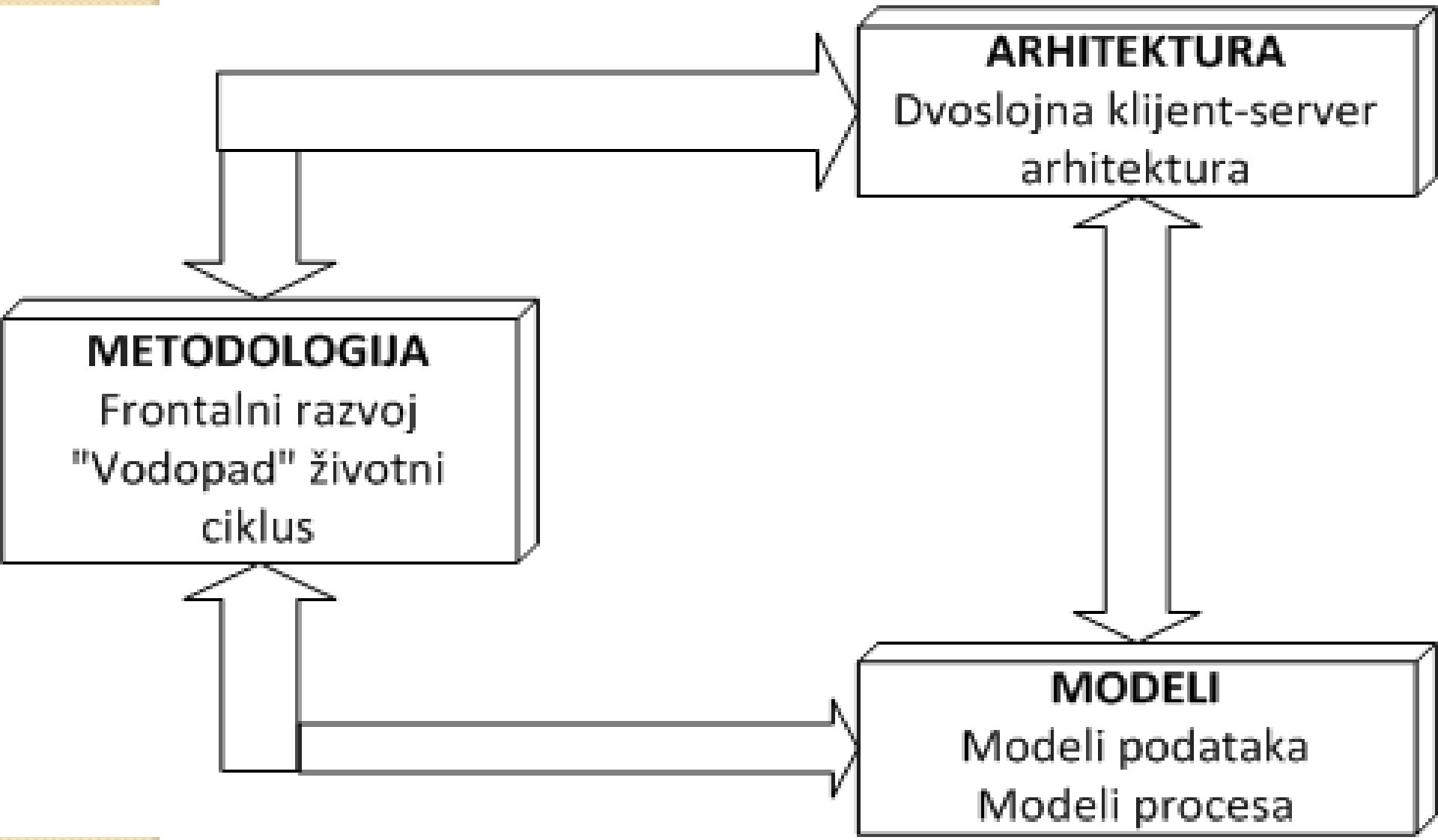
Uporedna analiza strukturnog i objektnog pristupa

Kao dva osnovna metodološka pristupa se izdvajaju **strukturni (konvencionalni, funkcionalni)** i **objektno-orijentisani pristup**. Ostali specifični i manje korišćeni pristupi su obično izvedeni iz ova dva osnovna pristupa. **Suštinska razlika** između ova dva pristupa **je u načinu kako se vrši dekompozicija sistema**.

U strukturnom pristupu **funkcionalna dekompozicija** je osnovni metodološki princip (paradigma) savladavanja složenosti.

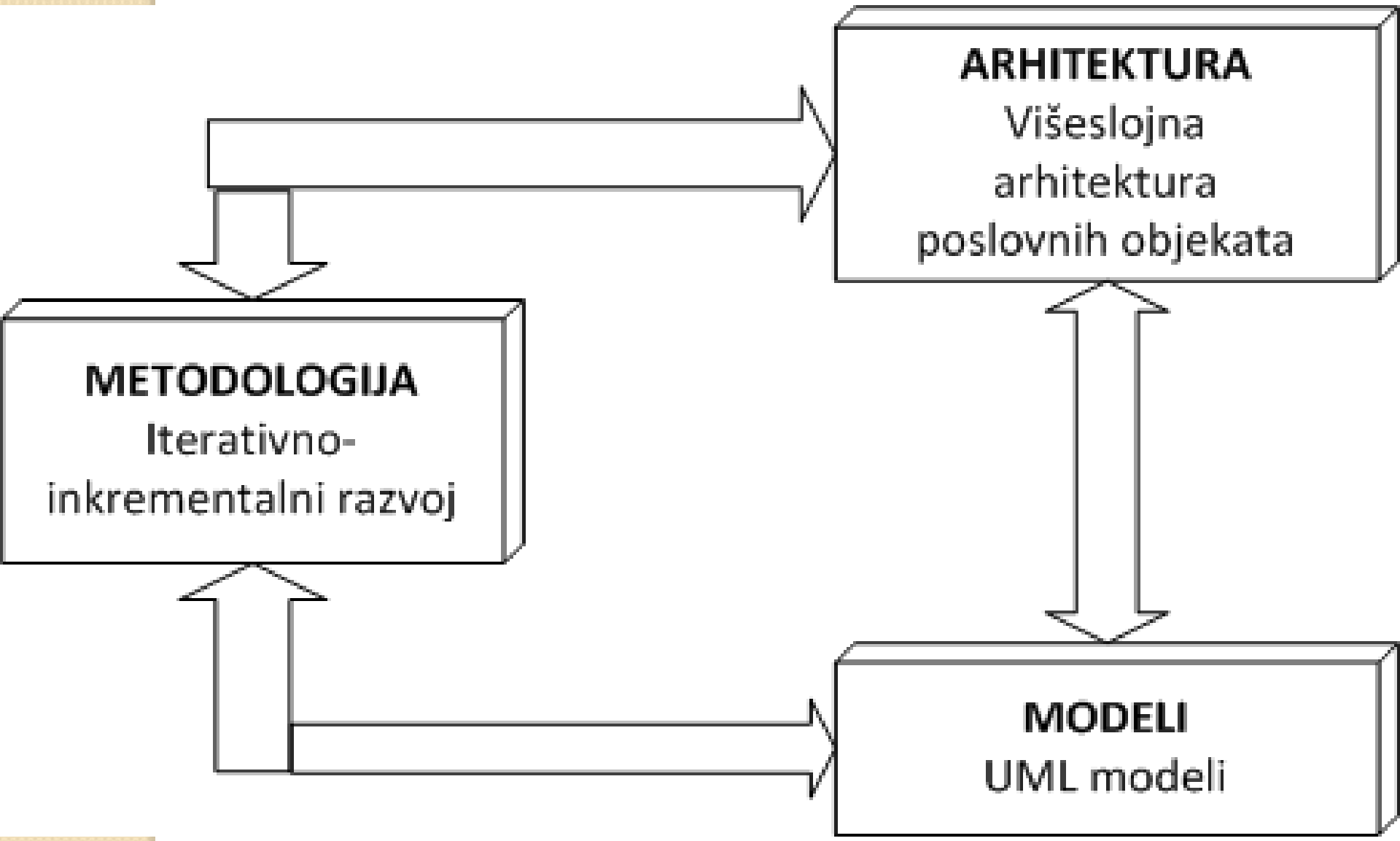
U OO pristupu osnovni metodološki princip savladavanje složenosti je **objektna dekompozicija**. Sistem se posmatra kao kolekcija međusobno povezanih objekata.

Uporedna analiza strukturnog i objektnog pristupa

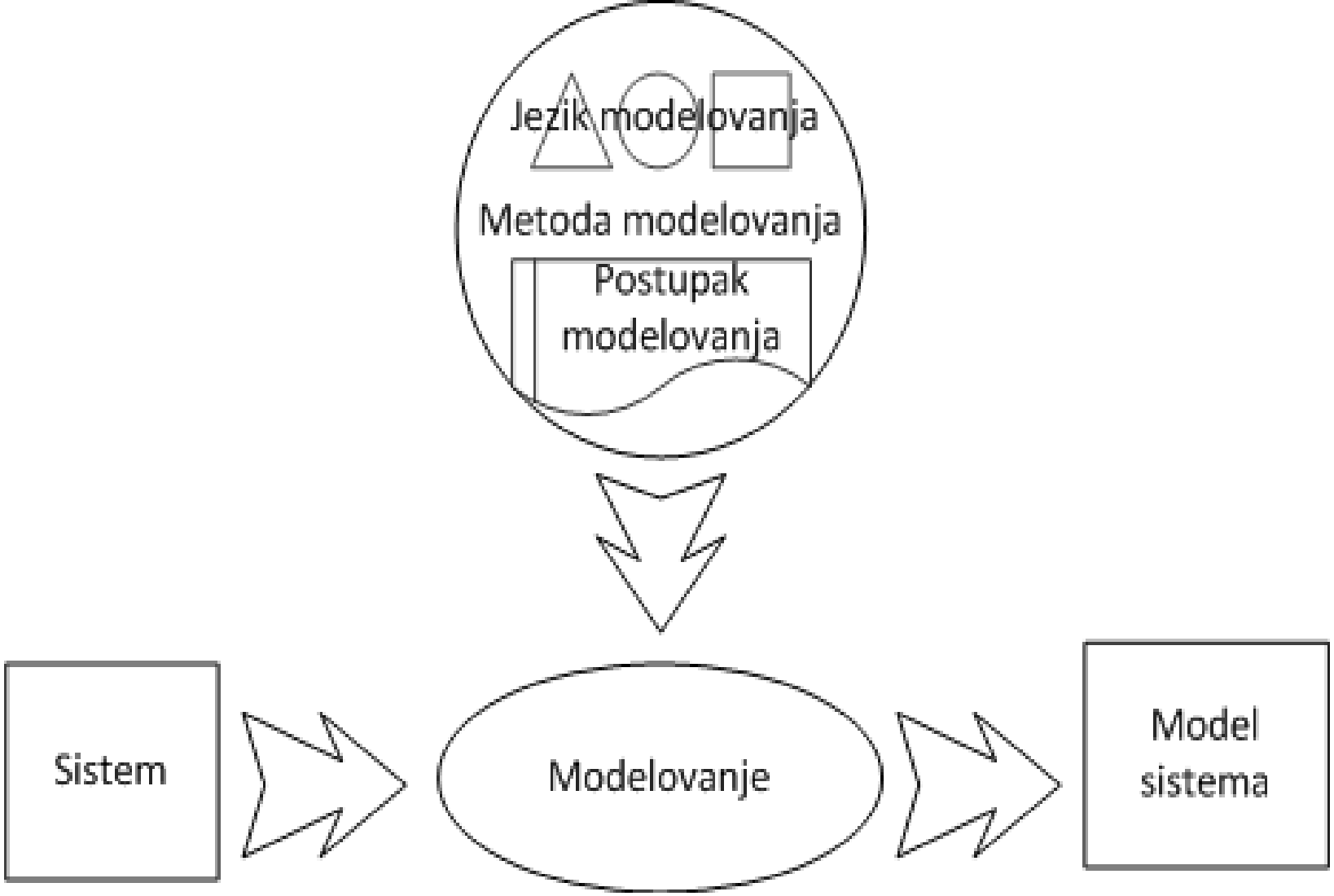


Strukturni pristup

Uporedna analiza strukturnog i objektnog pristupa



Objektno-orijentisani pristup



Odnos pojmova u modelovanju

1. **Metod** (opšta definicija) je **definisani postupak delovanja za postizanje određenog cilja** na nekom praktičnom ili teorijskom području.

Metod je **racionalni postupak kako doći do spoznaje** ili znanja, ili kako da se dokaže istina.

Jedna **potpuna metoda modelovanja** sastoji se iz:
- jezika modelovanja i
- postupka modelovanja, kao uputstva za korišćenje jezika u cilju uspešnog kreiranja modela.

- **Jezik modelovanja** definiše skup koncepata potrebnih za izgradnju modela, odnosno predstavlja alat za opis sistema modelom.

Jezikom modelovanja, kao i kod govornog jezika, definiše **sintaksa** i **semantika** koncepata koji se koriste i dodatno, **notacija**.
Sintaksa definiše koncepte i pravila jezika i opisana je gramatikom, dok **semantika** definiše značenje koncepata.
Notacija definiše za svaki koncept njegovu grafičku predstavu, odnosno simbol.

- Potrebno uputstvo za korišćenje jezika definiše se **postupkom modelovanja**, tako što se navodi **redosled koraka** za izgradnju modela, kao i **rezultat** koji se dobija primenom svakog od koraka (u vidu modela ili njegovog dela).

Korisne sugestije za modelovanje

- **Pokaži samo ono što moraš da pokažeš!**
- **Koristi dobro poznate notacije – ne izmišljaj!**
- **Reorganizuj velike dijagrame u nekoliko manjih!**
- **Pokušaj da praviš dijagrame na jednoj strani!**
- **Fokusiraj se na sadržaju – suštini dijagrama. Tek posle kad si siguran da si napravio ono što želiš, sredi izgled tvog dijagrama!**
- **Opiši dijagram sa komentarom!**
- **Koristi konzistentne i čitljive fontove!**

Metodologije (procesi) razvoja informacionih sistema

Termin softverska krize je uveden zbog **projektovanja IS bez primene odgovarajuće metodologije**.

Ideja softverskog inženjerstva je bila **uvođenje metodološkog, inženjerskog pristupa pri razvoju softvera** sa ciljem da se u zadanim vremenskim rokovima, preciznom primenom odgovarajućih modela i tehnika dođe do dovoljno kvalitetnog projekta, ali i do dovoljno kvalitetnog samog softvera.

Metodologije (procesi) razvoja informacionih sistema

Okosnicu softverskog inženjerstva predstavljaju **model životnog ciklusa IS** i **metodologija (proces) razvoja IS:**

1. **Način razbijanja na faze** se naziva **Model životnog ciklusa IS.** On polazi od činjenice da životni ciklus svakog proizvoda, odnosno životni vek, prolazi kroz iste faze. To znači da i razvoj IS treba da prati faznu logiku životnog ciklusa.

2. **Za izgradnju modela koriste se odgovarajuće metode. Skup specifičnih metoda, modela, tehnika i alata definisanih za svaku od faza** u razvoju IS predstavlja **metodologiju (proces) razvoja IS**.

Metodologije (procesi) razvoja informacionih sistema

Pod **metodologijom razvoja IS** se podrazumeva definisani **proces** razvoja softvera, gde se u različitim fazama primenjuju različiti standardni **modeli i metodi**.

Metodologije detaljnije opisuju faze razvoja i aktivnosti pojedine faze na najnižem potrebnom nivou detalja.

Metodologije (procesi) razvoja informacionih sistema

Da bi metodologija bila primenjiva za bilo koji sistem ona mora da bude opšta i da se njome precizno definišu: sve faze za realizaciju nekog IS, redosled faza, i specifični modeli i alati koji se koriste u svakoj od faza.

Prilikom izbora metodologije dozvoljena je sloboda, odnosno mogućnost definisanja sopstvene metodologije.

Životni ciklus

Koristeći se analogijom sa živim organizmima smatra se da svi proizvodi, čak i IS, imaju život konačnog trajanja.

Životni ciklus razvoja sistema (engl. systems development life-cycle) podrazumeva **niz faza i zadataka** (aktivnosti) na koje mogu biti **primenjene inženjerske metode** i gde se sve ove faze i zadaci zajedno **integrišu u metodologiju IS**.

Celovit pristup razvoju IS, kao i metodologije koje podržavaju ovakav pristup**, usko su povezani sa idejom životnog ciklusa** razvoja IS.

IS se u okviru životnog ciklusa mogu razvijati različitim pristupima i metodologijama.

Životni ciklus

Složenost razvoja IS savladava se razbijanjem celokupnog razvoja na faze. Način razbijanja na faze se naziva **Model životnog ciklusa IS.**

U praksi se sreću mnogi **modeli životnog ciklusa** razvoja sistema.

Modeli se razlikuju, kako po broju i nazivu faze, tako i po sadržaju aktivnosti koje se realizuju u okviru faza, metoda i principa koji se koriste.

Životni ciklus

Svaka pojedina aktivnost proizvodi skup rezultata.

Ciklus osigurava kontrolne tačke za praćenje progresa, procenu postignutih rezultata i donošenje odluka o daljim koracima.

Razvoj IS prema principima i modelima životnog ciklusa podrazumeva takav proces koji teče kroz niz sukcesivno procesnih faza, gde završetkom jedne faze započinje naredna i gde između susednih faza postoji snažna iterativna interakcija.

Životni ciklus

Različiti autori navode različite faze životnog ciklusa IS (razbijanje na faze razvoja IS), ali se u njegovom opštem toku većina slaže:

1.Priprema (“Planiranje rađanja");
2.Nastanak ("Rađanje");
3.Razvoj (“Život”);
4.Nestanak (“Umiranje”);

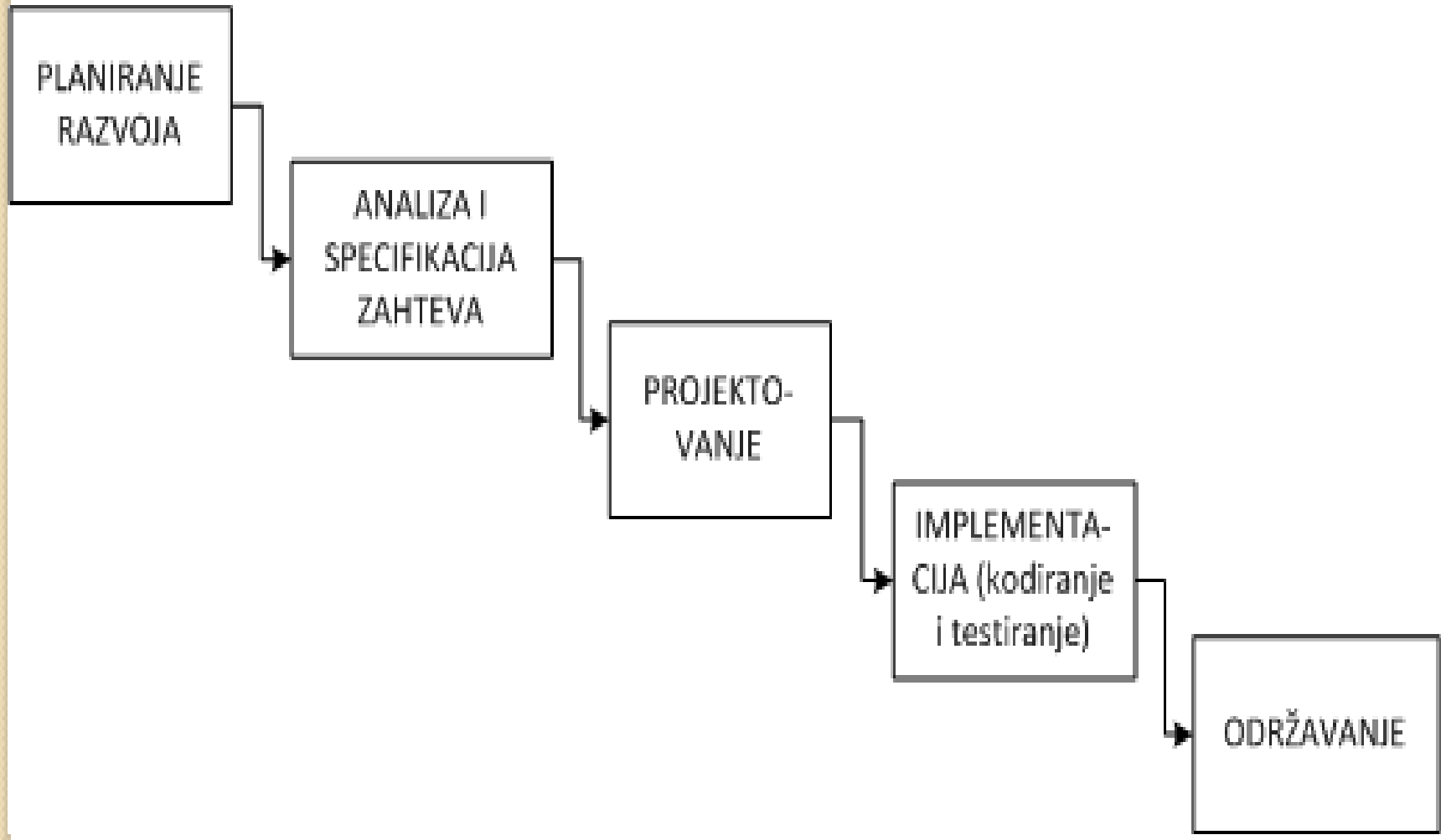
Bitno je da sve faze prate aktivnosti izrade odgovarajuće projektne, izvođačke dokumentacije i uputstva za upotrebu aplikacija, koja su sastavni deo razvoja IS.

Životni ciklus

Postoji podela na četiri osnovne grupe modela životnog ciklusa u odnosu na razvoj IS:

1. **Konvencionalni modeli**: striktno praćenje svih faza inženjerskog pristupa razvoju IS.
2. **Modeli zasnovani na brzom, agilnom razvoju**: što pre doći do kakvog takvog rešenja, pa ga onda usavršavati.
3. **Formalni (transformacioni) modeli**: definisanje formalnih modela i postupaka razvoja – formalna transformacija formalne specifikacije IS u implementaciju.
4. **Kombinovani:** konkretne metode i pristupi ne spadaju striktno ni u jednu grupu, već se kombinuju dva ili sva tri principa.

1. Konvencionalni model životniog ciklusa



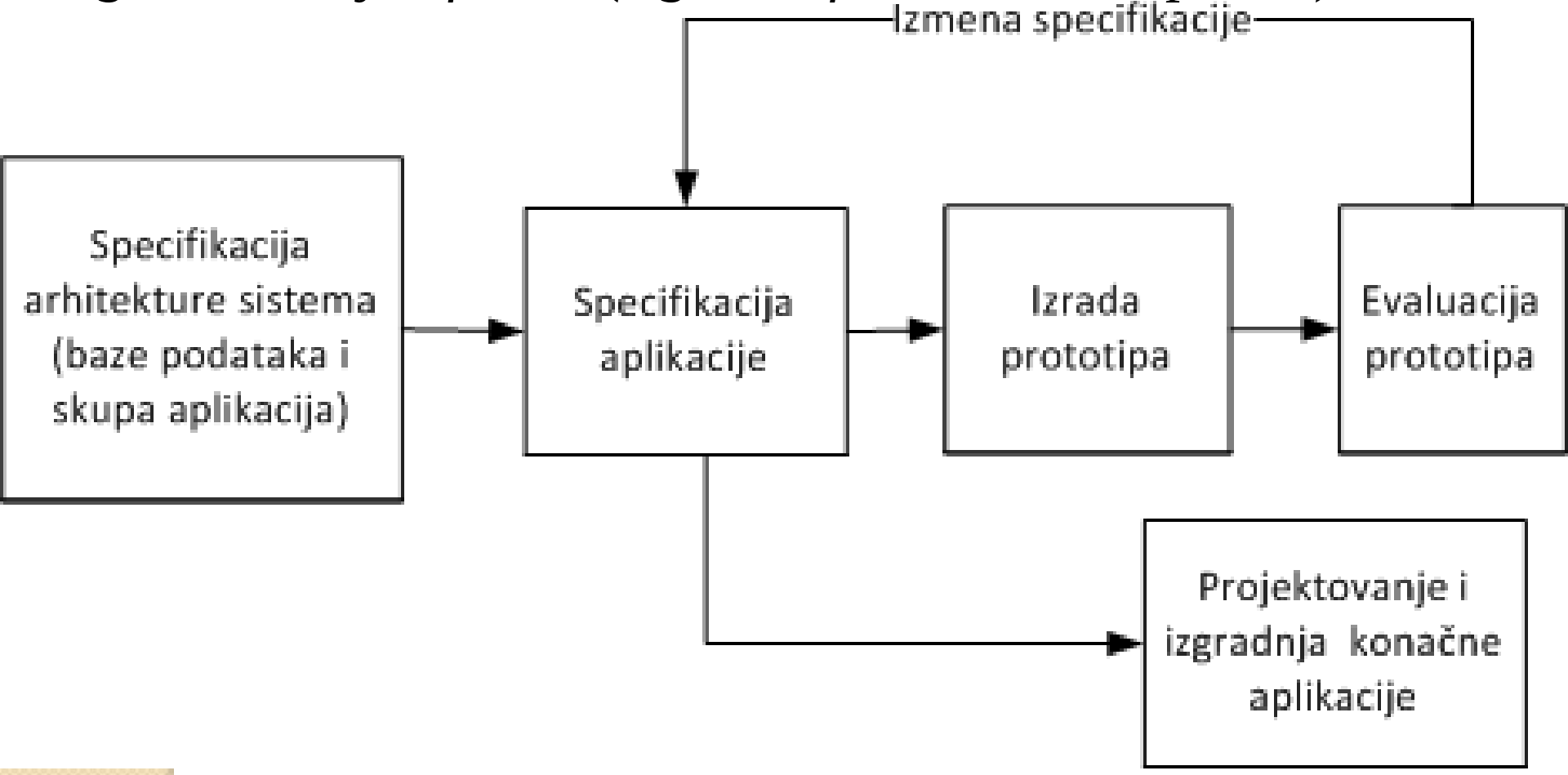
Vodopad životni ciklus

2. Modeli zasnovani na brzom, agilnom razvoju

Postoji podela brzog razvoja na:

1.Prototipski razvoj (Rapid prototyping)

2.Agilni razvoj softvera (Agile Software Development)



Prototipski razvoj

Prototipski razvoj (Rapid prototyping)

Prototipski razvoj dozvoljava projektantima da delimično eksperimentalno razvijaju sistem.

Posebno važna karakteristika prototipa je da **predstavlja eksperiment za komunikaciju sa korisnikom** i utvrđivanje njegovih zahteva na taj način.

Prototipovi koji se isključivo koriste za identifikaciju zahteva softvera nazivaju se brzi prototipovi i za njih se kaže da razvijaju pojedine delove sistema “brzo i prljavo”.

Agilni razvoj softvera

Cilj je kod koji radi, a ne perfektna dokumentacija. Više se vodi računa o organizaciji ljudi na projektu, a manje o metodologijama i alatima za razvoj.

Najviše uspeha u **agilnom razvoju** pokazuju timovi do **devet članova**. **Principi agilnog razvoja** su:

- Zadovoljavanje korisnika ranom i neprestanom isporukom upotrebljivog softvera;
- Stalne izmene zahteva korisnika;
- Kratki razvojni ciklusi, nekoliko nedelja do par meseci;
- Stalni interaktivan rad korisnika i inženjera;
- Učesnici su motivisani, ljubazni, talentovani, komunikativni “face to face”;
- Jednostavnost;
- Samoorganizovani timovi sa podelom odgovornosti;

Najpopularniji Agilni razvoj je **ekstremno programiranje** (*Extreme Programming* - XP). XP je **projektovan za male timove** koji treba da brzo razviju neki softver u okruženju sa stalno promenljivim zahtevima.

Agilni razvoj softvera (Agile Software Development)

Ekstremno programiranje se zasniva na sledećim principima:

- Planiranje: Korisnik daje procenu dobiti za realizaciju pojedinih zahteva, a programeri odgovarajuću procenu troškova i na osnovu toga se određuju zahtevi koji će biti realizovani odmah i koji će biti odloženi.
- Jednostavna realizacija: XP tim proizvodi jednostavne sisteme, a zatim ih učestalo menja i poboljšava.
- Metafora: XP timovi koriste zajednički sistem imena i opis sistema.
- Jednostavno projektovanje: podrazumeva da se svaki program pravi na najjednostavniji način.
- Testiranje: Stalno se vrši validacija sistema. Često se prvo definiše test, a onda se razvija softver koji treba da ga zadovolji. Korisnik definiše test za prihvatanje sistema.
- *Refactoring*: Softver se stalno poboljšava, ali se čuva njegova jednostavnost.

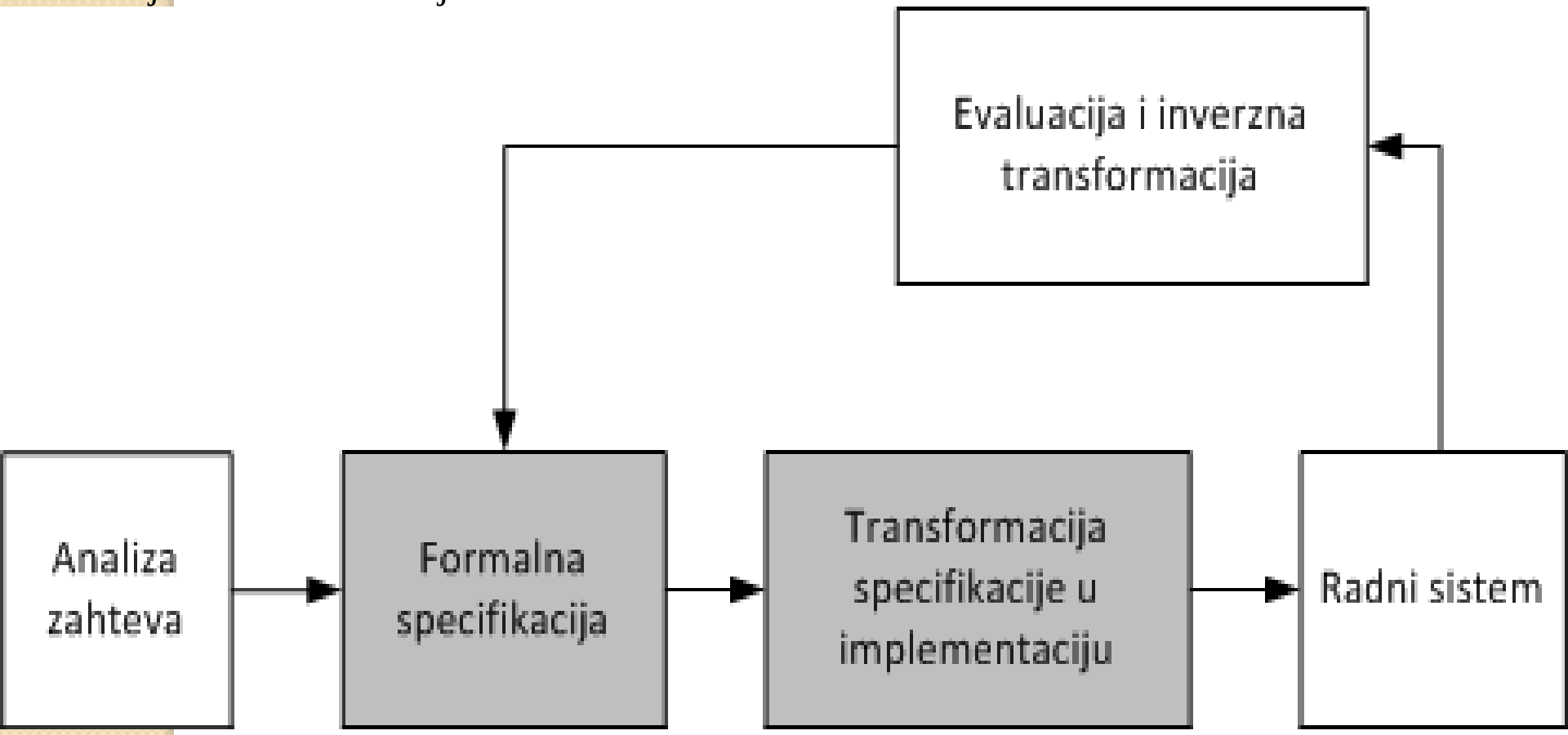
Agilni razvoj softvera (Agile Software Development)

Ekstremno programiranje se zasniva na sledećim principima:

- Rad u parovima: Dva programera razvijaju jedan kod kontrolišući jedan drugog i dobijaju bolji rezultati.
- Kolektivno vlasništvo: Ceo kod pripada svim programerima.
- Stalna integracija: Integracija razvijenih delova radi se više puta dnevno. To zahteva učešće svih programera i ubrzava proces razvoja.
- 40-časovna radna nedelja: Prekovremeni rad je izuzetak. Programeri treba da budu odmorni, zdravi i produktivni.
- Stalno prisustvo korisnika: Poboljšava se komunikacija, smanjuje potreba za prepiskom i dokumentacijom.
- Standardno kodiranje: Svi programeri treba da pišu kod na isti način.

3. Formalni (transformacioni) modeli

Ovo je jedan od najvažnijih kombinovanih razvoja, strukturnog i objektnog pristupa. Za razliku od inkrementalnog, gde se u svakoj sledećoj fazi razvoja dodaju bitno nova znanja odnosno karakteristike posmatranog sistema, **transformacioni pristup** zahteva da se svo znanje (što više znanja) o sistemu obuhvati već u fazi definisanja zahteva korisnika (u modelu zahteva), pa da se ona transformacijom prevode u druge modele, do implementacije, uz samo neophodno dodavanje novih informacija.



Transformacioni razvoj

3. Formalni (transformacioni) modeli

Karakteristike ovog razvoja su:

- Izvršni kod se dobija isključivo transformacijom formalne specifikacije u implementaciju;
- Ovakav pristup omogućava jednostavnu promenu implementacionog okruženja (platforme) - jednostavno adaptivno održavanje;
- Perfektivno održavanje sistema se značajno pojednostavljuje – nema dvostrukog održavanja (dokumentacije i koda);
- CASE alati sve više podržavaju ovakav pristup;
- Osnovna ideja i značajne prednosti ovoga pristupa su sve bolji jezici (modeli) za specifikaciju i CASE alati koji stalno unapređuju transformacioni pristup razvoja.

3. Formalni (transformacioni) modeli

Jedna od najpopularnijih metodologija koja primenjuje transformacioni pristup je: **Razvoj vođen modelom** (engl. Model Driven Development - MDD) u kome modeli imaju primarni značaj i predstavljaju osnovnu vrstu apstrakcije.

Za razliku od svih dosadašnjih metodoloških pristupa u kojima je glavni rezultat razvoja bio programski kod, a modeli uglavnom imali sekundarni značaj i služili kao pomoćno dokumentaciono sredstvo, u MDD-u je upravo obrnuto.

Modeli predstavljaju osnovni proizvod razvoja softvera, a programski kod se generiše na osnovu modela kojima se precizno i formalno opisuje sistem.

4. Kombinovani modeli

U mnogim situacijama modeli se mogu kombinovati tako da se postignu **prednosti od svih** na samo jednom projektu.

Ne treba biti isključiv u izboru određenog modela u razvoju IS.

Kombinovanjem modela, rezultat postignut u celini može biti povoljniji nego što bi to bio prosti zbir rezultata postignutih pojedinim modelima

Projektovanje informacionih sistema

HVALA !

dr Rade Matić