



SVEUČILIŠTE U ZAGREBU



Fakultet  
elektrotehnike i  
računarstva

# Razvoj komunikacijske programske podrške

Ak. god. 2021/2022

Održavanje programske podrške

# Održavanje softvera?



# Uzroci promjena programskog proizvoda

- Nove verzije prog. proizvoda
  - različiti uređaji, tehnologije
  - nove funkcionalnosti
  - ispravci identificiranih neispravnosti
  - prilagodba specifičnim korisničkim zahtjevima
- Na koje artefakte utječu navedene promjene?
- **Promjene se odnose i/ili utječu na**
  - zahtjeve
  - dizajn
  - kod
  - dokumentaciju

# Održavanje programskih proizvoda

Tijekom eksploatacije programskog proizvoda

- mijenjaju se/pojavljaju novi zahtjevi korisnika
  - mijenja se okolina programskog proizvoda
  - otkrivaju se neispravnosti koje se nisu otkrile tijekom faze testiranja.
- 
- Nemoguće je razviti programsku podršku koju nije potrebno održavati.
  - Stoga je potrebno razvijati programske proizvode pogodne za održavanje.
    - Cilj je smanjiti probleme i troškove održavanja

# Zastarjele komponente sustava (*legacy systems*)

- Sklopovlje (*obsolete mainframe hardware*)
- Programi podrške (*support software*) – često dobavljači više nisu na tržištu
- Aplikacijski programi (*application software*) – zastarjeli programski jezici, aplikacije
- Aplikacijski podaci - nepotpuni i nekonzistentni
- Poslovni procesi – ograničeni strukturom i funkcionalnostima PP-a.

# Dinamika programske evolucije

## 1. Lehmannov zakon - KONTINUIRANA PROMJENA

- Programski sustav koji se koristi u realnoj okolini mora se nužno mijenjati, inače postaje progresivno sve manje koristan u toj okolini.

## 2. Lehmannov zakon - POVEĆANJE SLOŽENOSTI

- Uvođenjem promjena u neki programski sustav, njegova struktura ima tendenciju degradacije.

## 3. Lehmannov zakon - EVOLUCIJA VELIKIH PROG. SUSTAVA

- Veliki sustavi imaju vlastitu dinamiku evolucije koja se uspostavlja u ranim fazama razvoja (održavanjem se ne može postići svaka željena promjena)

# Strategije promjene PP-a

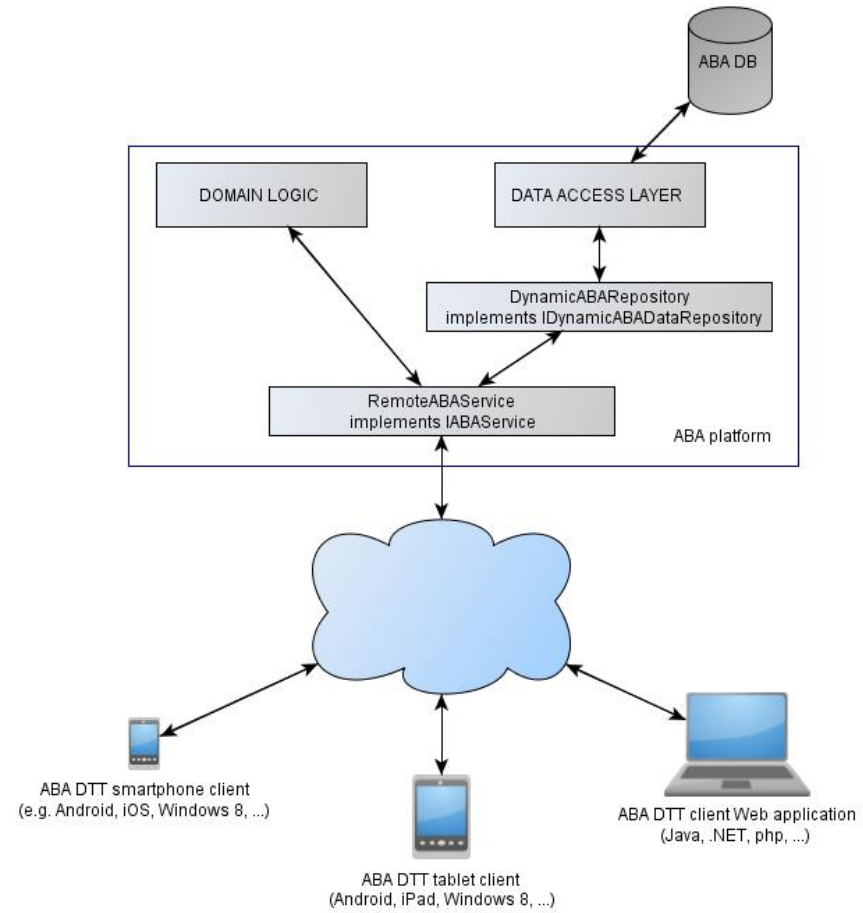
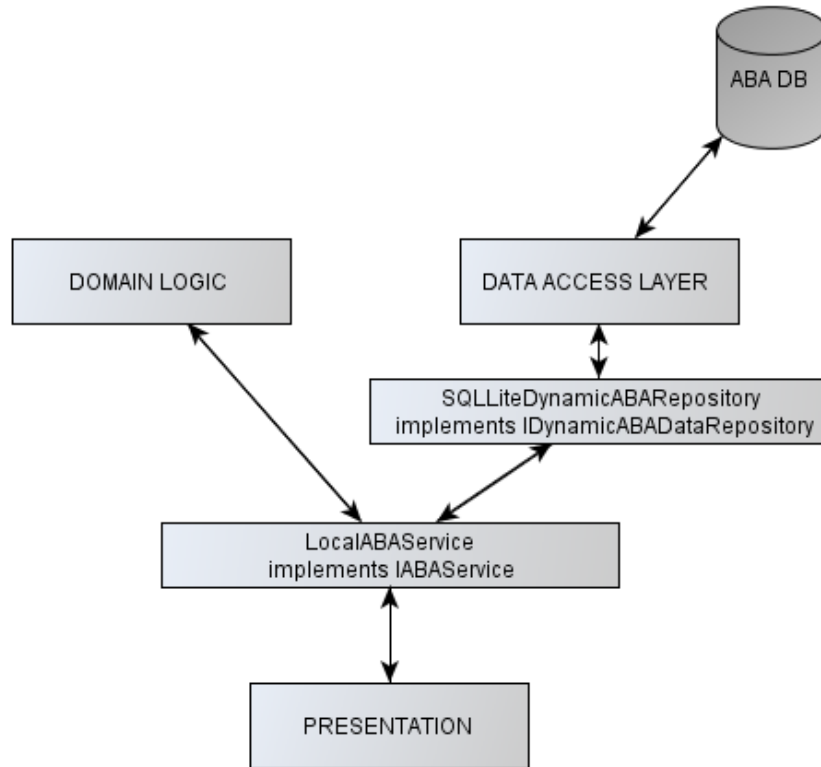
- Promjene funkcionalnosti
  - Unošenje promjena u PP zbog promjene zahtjeva, bez intencije promjene arhitekture (osim ako to nije posljedica novih zahtjeva)
- Promjena arhitekture
  - Općenito iz centralizirane u distribuiranu
- Re-inženjerstvo
  - Promjena strukture bez dodavanja novih funkcionalnosti
- Strategije mogu biti korištene pojedinačno ili skupno

# Evolucija arhitekture

- Potreba za promjenom centralizirane arhitekture u distribuiranu (npr. klijent-poslužitelj)
- Pokretači
  - troškovi sklopovlja
  - potreba za korištenjem grafičkih sučelja
  - distribuirani pristup sustavu



# Evolucija arhitekture - primjer



# Pitanje

- Kada počinje proces održavanja?
- Kada treba početi voditi računa o procesu održavanja?

# Održavanje programskih proizvoda (2)



*Aktivnosti neophodne za osiguravanje efikasne podrške programskom proizvodu*

## **Aktivnosti prije isporuke**

- planiranje akcija koje će se poduzeti nakon isporuke
- pogodnost za održavanje
- logistika održavanja

## **Aktivnosti nakon isporuke**

- modifikacija programa
- obuka korisnika
- organiziranje službe za podršku korisniku

# Povijest

- 1960 – 1970
  - Uključivanje održavanja u vodopadni model
  - Percepcija da se aktivnosti poslije isporuke svode na ispravljanje neispravnosti i manje promjene
  - Nisu uključene promjene funkcionalnosti
- 1970
  - Lehmanovi zakoni evolucije
  - Naglašava se potreba kontinuirane evolucije s obzirom na promjene softverske okoline
- Kasne 1970 – 1980
  - Inicijalni modeli procesa koji uključuju zahtjeve za promjenom
- 1990
  - Općenito prihvaćanje evolucije PP-a
  - Novi modeli procesa (evolucijski, spiralni, agilni)

# Odnos procesa održavanja i razvoja PP

## ◆ SW sustav

### Proces razvoja SW

Potrebno je uključiti održavanje u rane faze životnog ciklusa programskog proizvoda!

- Implementirani zahtjevi
- Ispravan rad



Korisnik



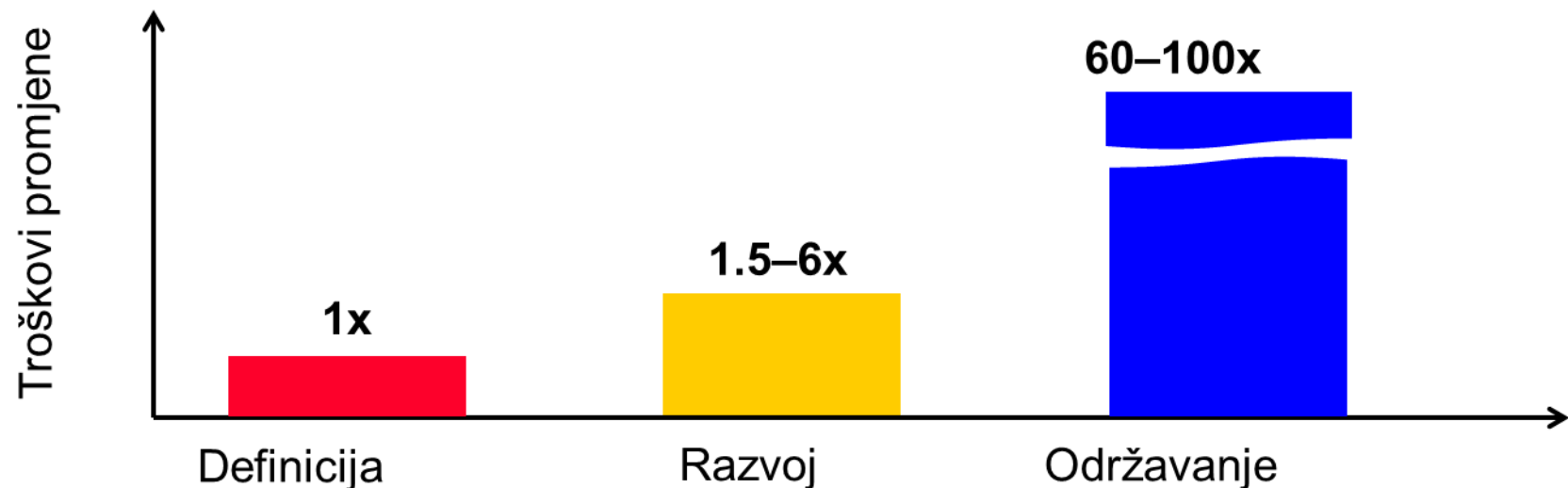
- Programski kod
- Dokumentacija

Održavanje SW



# Značajke održavanja

- Održavanje je najduža i najskuplja faza u životnom ciklusu programskog proizvoda.
- Istraživanja
  - 25%-33% ukupnih troškova razvoj
  - 67% implementiranje PP u stvarnu okolinu održavanje.



# Značajke održavanja (2)

- Održavanje je potrebno kako bi se osiguralo da sustav kontinuirano zadovoljava zahtjeve korisnika.
- Ciljevi održavanja programskih proizvoda:
  - ispravljanje grešaka,
  - izrada sučelja prema drugim sustavima,
  - ugradnja različitih poboljšanja,
  - izvedba neophodnih promjena sustava,
  - poboljšanje dizajna,
  - pretvorba programa tako da se može koristiti drugo sklopovlje, drugi programi i značajke sustava.

# Značajke održavanja (3)

- Osnovni aspekti na koje se održavanje fokusira jesu
  - održavanje kontrole nad svakodnevnim funkcioniranjem sustava,
  - održavanje kontrole nad modifikacijama sustava,
  - poboljšanje postojećih funkcija,
  - prevencija performansi sustava od degradacije na neprihvatljivu razinu.





# Značajke održavanja (4)

- Održavanje otežava
  - stav da je “održavanje akcija koja slijedi samo nakon isporuke proizvoda”,
  - nedostatak pažnje tijekom procesa razvoja prema elementima PP koji bi kasnije olakšali održavanje,
  - nerazumijevanja korisnika da je održavanje akcija koja ponekad zahtijeva velike strukturalne promjene



# Kategorije održavanja

## Korektivno

- Pronalaženje i ispravljanje pogrešaka pronađenih u programu nakon isporuke, greške koje nisu otkrivene tijekom faze testiranja.
- 20% ukupnih troškova održavanja



# Kategorije održavanja

## Adaptivno

- Održavanje inicirano promjenama u programskoj okolini koje uključuju novo sklopovlje, novi OS ili nove verzije postojećeg OS-a, nova periferna oprema itd.
- 25% ukupnih troškova održavanja



iOS 6



# Kategorije održavanja

## Perfektivno

- Zadovoljavanje novih i modificiranih potreba korisnika.
- Promjena postojećih ili dodavanje novih funkcionalnosti u programsku podršku.
- 55% ukupnih troškova održavanja
- Uvijek inicirano od strane korisnika.



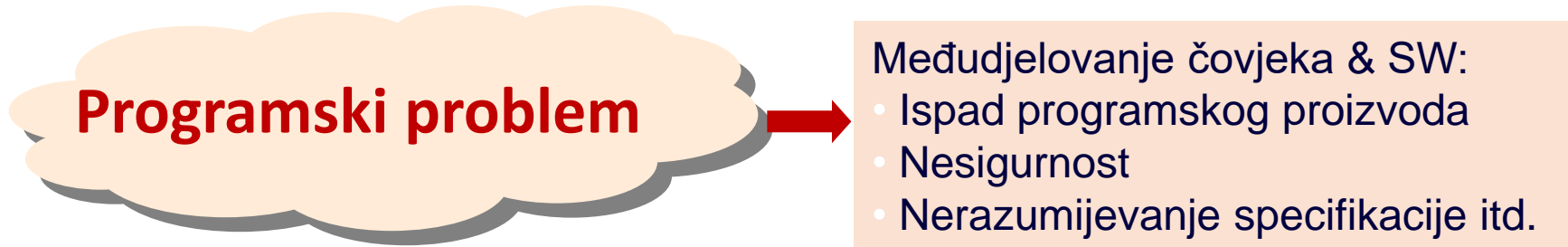
# Kategorije održavanja

## Preventivno

- Sprečavanje eventualnih problema prije nego se pojave.
- Sprečavanje degradacije performansi programskog proizvoda na neprihvatljivu razinu.
- 5% ukupnih troškova održavanja.



# Korektivno održavanje



**Ispad**

Izlazno stanje definirano specifikacijom  
≠ stvarni rezultati izvođenja

**Neispravnost**

Dio programskog koda koji pod određenim  
uvjetima uzrokuje ispad programa

# Održavatelji

- Ugovorima između razvijatelja programskog proizvoda i korisnika definira se tko će vršiti održavanje.

## 1. Organizacija/odjel za razvoj programskog proizvoda

- + osobe koje su se bavile razvojem nekog programskog proizvoda imaju najbolje znanje o njemu; nema potrebe za detaljnom dokumentacijom; nema potrebe za uspostavom formalnog komunikacijskog sustava između razvoja i održavanja, ...
- preveliko “dotjerivanje” (ono što se nije stiglo pri razvoju), ...

# Održavatelji (2)

## 2. Nezavisna organizacija/odjel za održavanje

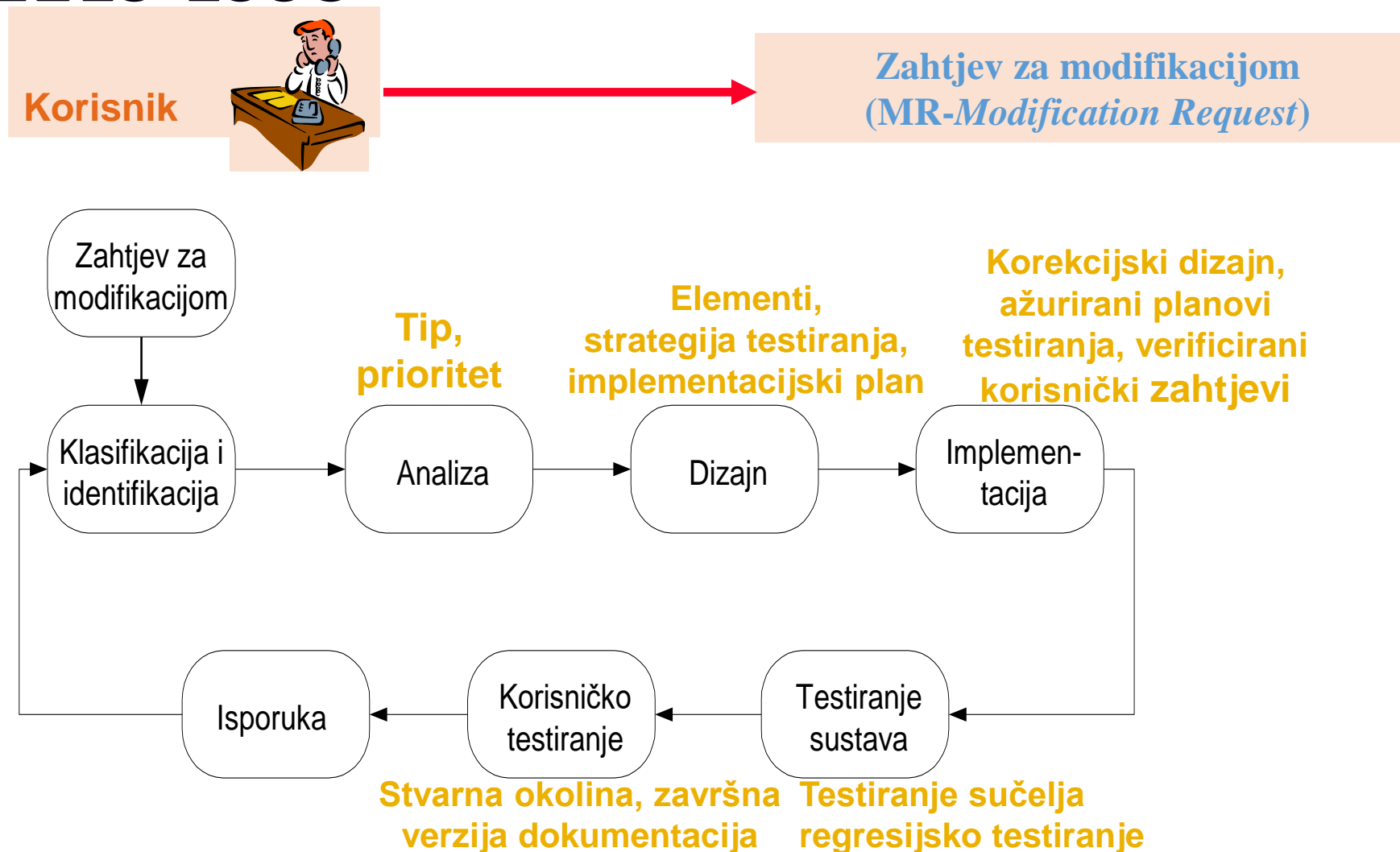
- + veća pažnja posvećuje se stvaranju bolje dokumentacije; uspostavlja se formalna procedura za prebacivanje programskog proizvoda iz faze razvoja u fazu održavanja; osoblje koje vrši održavanje ima mogućnost prepoznavanja prednosti i mana programskog proizvoda, ...
- prebacivanje sustava iz razvoja u održavanja može biti sporo; pojavljuju se problemi refundiranja troškova održavanja; potrebno je određeno vrijeme da organizacija/odjel za održavanje stekne znanja o novom programskom proizvodu koji je preuzet, ...



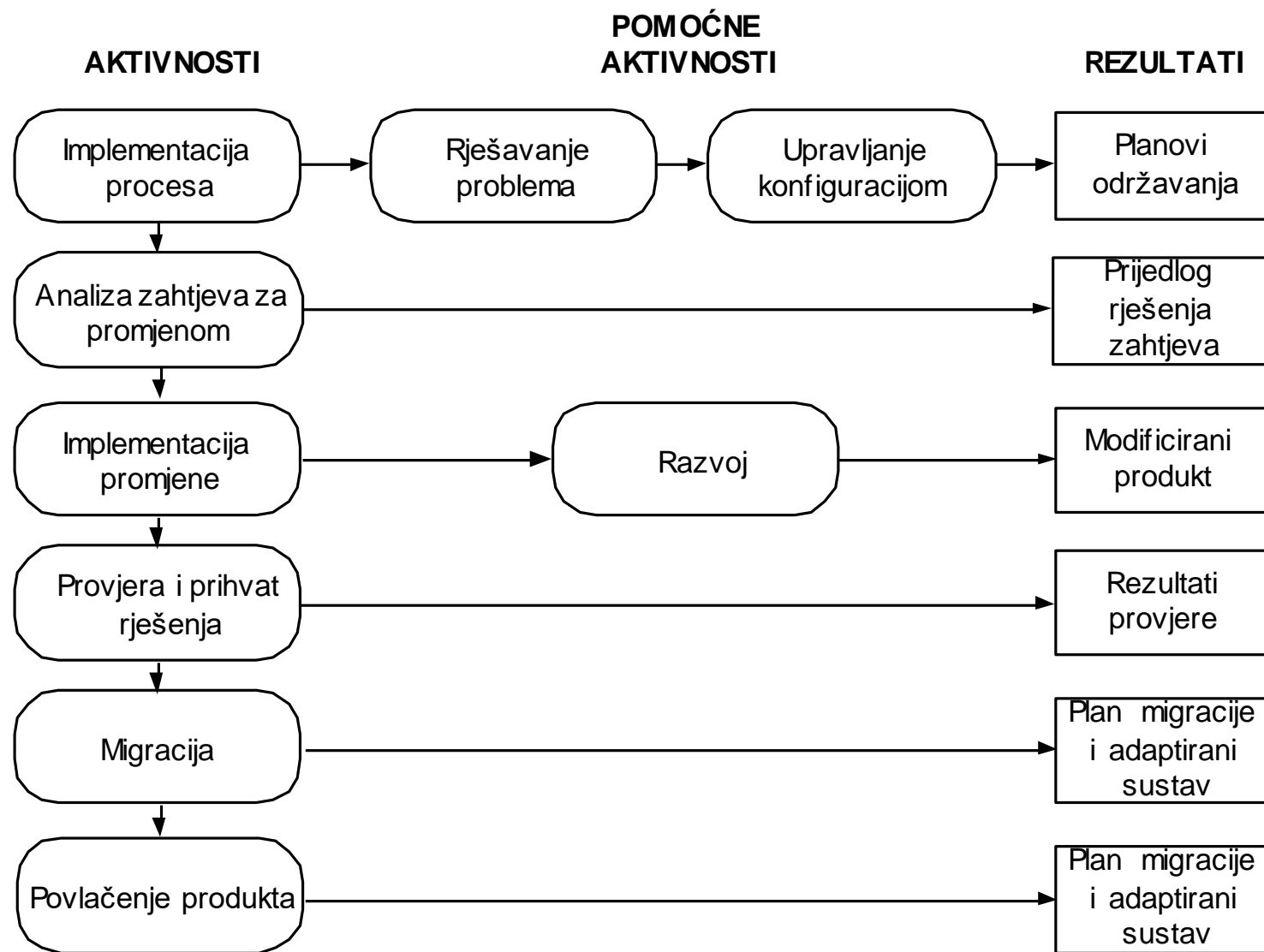
# Problemi

- Ograničeno razumijevanje:
  - loša dokumentacija
  - 40-60% troškova održavanja (više što je PP stariji)
- Problemi testiranja
  - teško odrediti razinu napora testiranja, ažurnost skupa regresijskih testova
- Analiza utjecaja
- Procjena troškova i napora (mnogo nepoznanica)
- Degradacija arhitekture, povećana entropija

# Model održavanja prema standardu IEEE 1219 1993




# Model održavanja prema standardu ISO/IEC 12207



# Aktivnosti održavanja

- **Nestrukturirano održavanje**

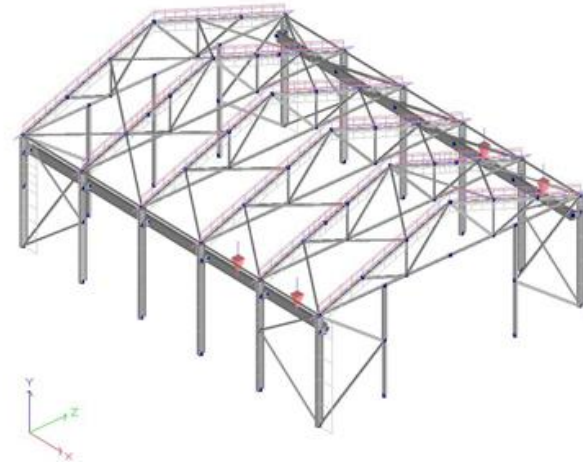
- dostupan je jedino kôd PP 
- dokumentacije nema ili je oskudna
- analiza koda      konfiguracija, globalne strukture podataka, sučelja, performanse PP, ograničenja dizajna
- teško procijeniti utjecaj promjena i nemoguće **regresijsko testiranje** (= dio testiranja sustava kojim se provjerava je li unošenje promjena prouzročilo neispravnosti unutar programskog proizvoda)



# Aktivnosti održavanja (2)

- **Strukturirano održavanje**

- PP je razvijen primjenom metodologija programskog inženjerstva
- poznata konfiguracija PP, dostupna dokumentacija
- analiza dokumentacije o dizajnu      određivanje značajki PP (struktura, performanse, sučelja)
- planiranje promjena, primjena regresijskog



# Troškovi održavanja

## Napori održavanja

### Produktivne aktivnosti

(analiza i procjena,  
modifikacije dizajna,  
kodiranje, ...)

### Aktivnosti prikupljanja informacija

(razumijevanje koda, interpretiranje  
struktura podataka, značajke sučelja, ...)

$$M = p + Ke^{(c-d)}$$

$M$  ... ukupni naponi održavanja

$p$  ... naponi uloženi u produktivne aktivnosti

$K$  ... empirijska konstanta

$c$  ... mjerila složenosti zbog lošeg dizajna ili dokumentacije



# Performanse procesa održavanja



# Perfektivno održavanje

- Razlozi velikih troškova
  - (usporedba: dodavanje novih funkcionalnosti prog. sustavu nakon isporuke - originalni razvoj prog. sustava iste funkcionalnosti)
  - Održavatelji - nedostatak iskustva s aplikacijom
  - Nestrukturirani razvoj (efikasnost, bez prog. inž.)
  - Uvođenje promjene - nove neispravnosti - novi zahtjevi za modifikacijom
  - Uvođenjem promjene - tendencija degradacije strukture sustava (sustav teže razumjeti i teže unijeti promjenu)
  - Gubitak veze između određene verzije i dokumentacije



# Perfektivno održavanje

- (Razlozi velikih troškova)
  - Stabilnost tima
    - Troškovi manji ako je tim stabilan kroz neko vrijeme
  - Ugovorne odgovornosti
    - Ako ne postoji može se razviti kod k



# Dokumentacija važna za održavanje

- Dokumentacija o prog. zahtjevima
- Dokumentacija o arhitekturi
- Programski kod
- Dokumenti o testiranju
- Dokumenti o prethodnim zahvatima održavanja
- **Strukturirana:** organizirana u čitljive odvojene cjeline - *poglavlja* (ne gubi se vrijeme na loš izgled i sadržaj), gdje je moguće treba proizvesti dokumentaciju s *CASE pomagalima*
- **Predložak** dokumentacije koji se popunjava tijekom programskih procesa

# Naslijeđeni sustavi

- *Legacy systems*
- Naslijeđeni sustavi su stari sustavi koji postaju teški za održavanje
  - Akumulacija promjena narušava modularnost originalne arhitekture
  - Dokumentacija nije održavana i zastarjela
  - Djelomična zastarjelost tehnologije
- Dvije komplementarne tehnike za kontinuiranu evoluciju naslijeđenih sustava
  - Reverzno inženjerstvo
  - Reinženjerstvo

# Reverzno inženjerstvo

- *Reverse Engineering*
- Proces analize nekog programskog sustava kako bi se identificirale komponente sustava i njihovi međuodnosi s ciljem kreiranja prikaza sustava u formi na višoj razini apstrakcije.
  - Ova tehnika ne mijenja sustav i ne kreira novi sustav.
  - Primjer je obnavljanje dokumentacije i dizajna.



# Programsko reinženjerstvo

- *Re-engineering*
- Kombinacija reverznog inženjerstva i restrukturiranja programskih podataka, arhitekture i logike
- Cilj: razvoj PP iste funkcionalnosti i veće kvalitete
  - Ponovna implementacija cijelog ili dijela naslijeđenog sustava kako bi postao pogodan za održavanje
  - Pogodno kad neki dijelovi sustava zahtijevaju česte zahvate održavanja
  - Uz sustav treba se ažurirati i dokumentacija

# Pogodnost za održavanje

- *Maintainability*
- Mjera za definiranje jednostavnosti (lakoće) s kojom se neki programski proizvod može razumjeti, ispraviti, prilagoditi ili poboljšati.



## Pogodnost za održavanje

### Upravljački faktori

- Metodologija razvoja (dizajn, kodiranje, testiranje)
- Razvojna okolina (osoblje, struktura PP, programski jezici, dokumentacija, dostupnost test sekvenci, ugrađen *debugging* )

### Kvantitativna mjerenja

- Mjere koje se odnose na vrijeme

# Mjerenja i metrika - pojmovi, razlike

- Mjerenja

- Kvantitativne informacije o programskom procesu
- Procjena prednosti i nedostataka procesa
- Kontekst odnosa procesa i njegovih rezultata



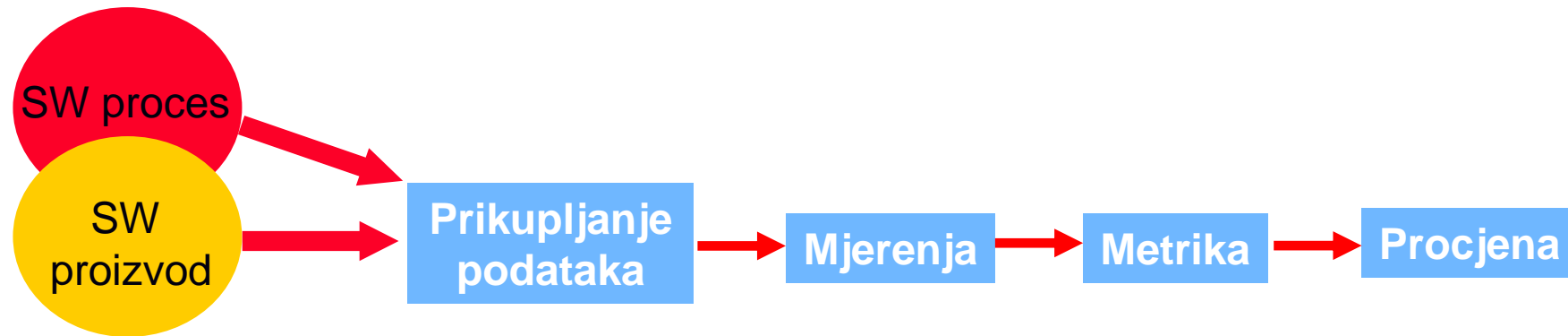
- Metrika

- Određuje u kojoj mjeri programski proces sadrži zahtijevana svojstva





# Mjerenja i metrika u procesima razvoja i održavanja softvera



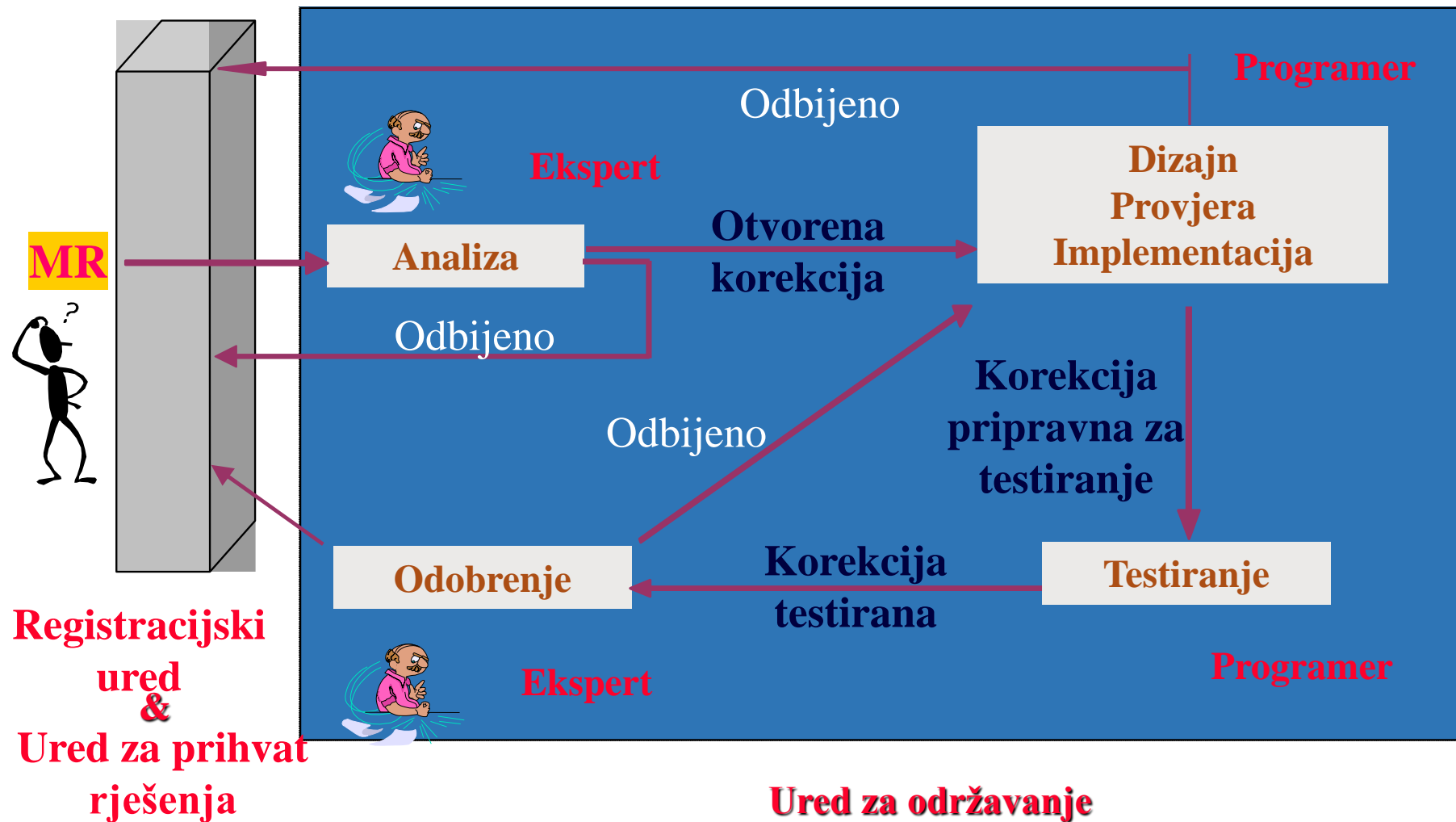
# Metrika procesa održavanja

- Procesna metrika za procjenu održavanja
  - Broj zahtjeva korektivnog održavanja
  - Prosječno vrijeme provođenja analize utjecaja
  - Prosječno vrijeme implementacija zahtjeva za promjenom
  - Prosječan broj neimplementiranih zahtjeva za promjenom
- Ako jedan od faktora počne rasti – smanjena pogodnost za održavanje.

# Zahtjev za modifikacijom

- *Modification Request (MR)*
- Standardizirana forma
- Generiran od strane razvijatelja PP
- Popunjava korisnik koji zahtijeva aktivnosti održavanja
- Opis svih uvjeta koji su doveli do pojave neispravnosti
- Specifikacija promjena
- MR dokument služi kao osnova za planiranje aktivnosti održavanja

# Studijski primjer procesa održavanja



# Dodatna literatura

1. Car, Željka. Software Maintenance Process Modeling and Analysis. *Managing Corporate Information System Evolution and Maintenance*, London : Idea Group Publishing, 2004. pp. 376. - dostupno kod Ž. Car
2. SWEBOK – SW Body of Knowledge: Chapter SW Maintenance.  
<https://www.computer.org/education/bodies-of-knowledge/software-engineering>