23. POUZDANOST SOFTVERA, METODE, TEHNIKE I METRIKE

Pouzdanost je jedan od ključnih faktora kvaliteta i zato pouzdanost isporučenog softvera zavisi od kvaliteta svih procesa i rezultata rada u svim životnim fazama razvoja softvera.

Pouzdanost softvera se definiše kao niz karakteristika, **tj. atributa koji predstavljaju mogućnost softvera da održi potreban nivo performansi pod određenim uslovima u određenom vremenskom periodu.**

Kod softvera ne postoje pojmovi kao što su: trošenje, starenje, zamor materijala, pa je nepouzdanost posledica grešaka u zahtevima, projektovanju i implementaciji.

Karakteristike pouzdanosti softvera su:

- **Zrelost** (Engl. "Maturity") - osobina softvera koje se odnosi na **frekvenciju grešaka** zbog nedostataka u softveru;
- **Otpornost prema greškama** (Engl. "Fault tolerance") - osobina softvera da održi određeni **nivo performansi** u slučaju greške;
- **Oporavljivost** (Engl. "Recoverability") - svojstvo softvera da **ponovo uspostavi svoj nivo performansi i povrati podatke** na koje je direktno uticao u slučaju greške.

Izgradnja visoko pouzdanog softvera zavisi od učešća karalteristika pouzdanosti i kvaliteta u svakoj fazi životnog ciklusa razvoja sa naglaskom na prevenciju grešaka, specijalno u ranim fazama.

Da bi se ovi atributi mogli meriti potrebno je bilo definisati softverske metrike za svaku razvojnu fazu. Standard IEEE 982.1-1988 definiše upravljanje pouzdanošću softvera kao proces optimizacije softvera kroz tri aktivnosti u kontekstu ograničenja u projektu kao što su resursi, vremena i performanse:

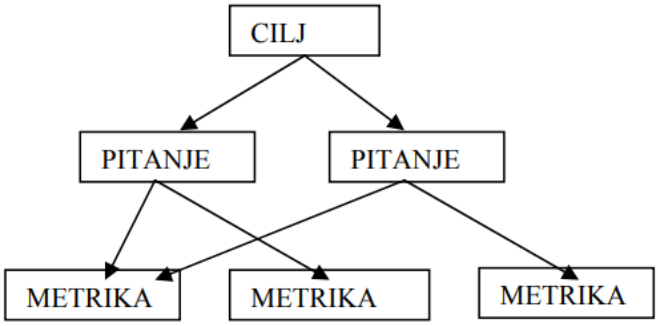
- Prevencija grešaka;
- Detekcija i otklanjanje grešaka;
- Merenja za maksimizaciju pouzdanosti prve dve aktivnosti.

Basili i Rombach su 1988. predstavili **GQM, tj. Engl. Goal-Question-Metric (cilj-pitanje-metrika) paradigmu. GQM postavlja ciljeve pre bilo koje aktivnosti merenja atributa pouzdanosti softvera**. Da bi merenje bilo efikasno mora biti fokusirano na specifične ciljeve i primenjeno u svim životnim ciklusima razvoja softverskog proizvoda, interpretirano na razumevanju organizacionog konteksta, okruženja i ciljeva softvera i njegove upotrebe. To znači da merenje mora biti definisano u top-down (od vrha prema dnu) stilu i mora biti fokusirano i bazirano na ciljevima i modelima.

GQM model ima tri nivoa:

1. **Konceptualni nivo (GOAL)** - Cilj se definiše za objekat, vodeći računa o različitim modelima kvaliteta i različitim tačkama gledišta. Objekti mjerenja su: proizvodi koji nastaju u životnom ciklusu softvera npr. specifikacije, modeli, programi, testni podaci i sl., zatim procesi koji obuvataju aktivnosti vezane za vreme, resursi upotrebljeni u procesu da bi se dobio proizvod: osoblje, hardver, softver, radni prostor.
2. **Operativni nivo (QUESTION)** - Niz pitanja koja se upotrebljavaju da specificiraju put za postizanje željenog cilja. Pitanja treba da karakterišu objekte (proizvodi, procesi, resursi).
3. **Kvantitativni nivo (METRIC) -** Skup podataka pridruženih svakom pitanju da bi se dobio odgovor u kvantitativnoj formi. Podaci mogu biti objektivni i subjektivni.

GQM model ima hijerarhijsku strukturu prikazanu na sledećoj slici:



Metrike treba da odgovore na tri kompleksna pitanja vezana za zahteve, programski kod i testiranje, a u cilju poboljšanja pouzdanosti i pronalaženje modula sklonim greškama:

1. **Kakav je kvalitet zahteva (dokumenta)?** Broj linija teksta, komandne reči i fraze koje slede zahteve na nižem nivou, direktive - reference na slike, tabele i napomene, slabe fraze - neizvesnost i višestruka interpretacija, nekompletnost iskaza, opcije.
2. **Kakav je kvalitet dizajna i koda?** Veličina koda, kompleksnost koda, funkcionalne tačke, modularnost, ponovna upoteba koda.
3. **Koliki je broj pronađenih grešaka? V**reme između grešaka, vreme nastajanja greške, vreme otklanjanja greške, frekvencija događanja grešaka, predviđanje broja preostalih grešaka.