**1. Definirajte pojam softverskog proizvoda i pojam životnog ciklusa softverskog proizvoda**

Softverski proizvod je skup računalnih programa i odgovarajuće dokumentacije koji se

isporučuje korisniku.

Životni ciklus softverskog proizvoda čine svi procesi, aktivnosti i zadaci uključeni u

koncept, definiciju, razvoj, funkcioniranje i održavanje softverskog, proizvoda.

U životnom ciklusu softverskog proizvoda mogu se razlučiti dvije osnovne faze:

razvojna faza tijekom koje se softverski proizvod inicijalno razvija i

eksploatacijska faza, tijekom koje se softverski proizvod praktično koristi.

Granicu između te dvije faze predstavlja isporuka softverskog proizvoda korisniku.

**2. Koji su osnovni sudionici softverskih procesa?**

Sudionici softverskih procesa su sve osobe i organizacije uključene u aktivnosti vezane uz

razvoj i eksploataciju softverskih proizvoda:

- kupac - osoba ili organizacija kojoj je softverski proizvod namijenjen i koje obično (ali

ne i nužno) određuju zahtjeve koje softverski proizvod mora zadovoljiti,

- korisnik - osoba ili organizacija koja izravno koristi softverski proizvod,

- dobavljač - organizacija koja sklapa ugovor s kupcem o isporuci softverskog

proizvoda (može bili i razvijatelj),

- održavatelj - osoba ili organizacija koja provodi aktivnosti održavanja softverskog

proizvoda.

**3. Koje su prednosti i nedostaci evolucijskog razvoja?**

Prednost evolucijskog razvoja je što se specifikacija može razvijati inkrementalno prateći razvoj korisnikovog razumijevanja problema. Nedostatci ovog pristupa uključuju nedovoljan uvid u proces i nemogućnost praćenja brzog razvoja kvalitetnom dokumentacijom,(sa stajališta vođenja projekta), te slaba strukturiranost sustava zbog

kontinuiranog procesa izmjena.

**4. Koje forme može imati prototipski model?**

Može imati 3 različite forme.

- Papirnati prototip ili model na osobnom računalu koji predstavlja interakciju čovjeka i

stroja na način razumljiv korisniku.

- Radni prototip koji implementira neki podskup funkcija softvera koji treba razviti.

- Postojeći program koji izvodi dio ili sve zahtijevane finkcije, ali mu ostala svojstva ne

zadovoljavaju postavljane zahtjeve.

**5. Koje su glavne aktivnosti u procesu inženjerstva zahtjeva?**

1. Studija izvodljivosti (feasibility study) predstavlja procjenu o mogućnosti

zadovoljenja korisničkih zahtjeva raspoloživim softverskim i hardverskim

tehnologijama. Procjenjuje se ekonomičnost predloženog sustava i odnos prema

financijskim mogućnostima. Studija izvodljivosti treba biti relativno jeftina i brza.

Rezultat treba omogućiti odluku o provedbi detaljnije analize.

2. Izvođenje i analiza zahtjeva (requirements elicitation and analysis) obuhvaća

izvođenje zahtjeva na temelju promatranja postojećih sustava, razgovora s

potencijalnim korisnicima, analiza zadataka i drugog. Ovaj proces može uključivati

razvoj jednog ili više modela sustava ili prototipova.

3. Specifikacija zahtjeva (requirements specification) je aktivnost koja obuhvaća

prevođenje informacija prikupljenih tijekom analitičkih aktivnosti u dokument koji

definira skup zahtjeva. Ovdje su obuhvaćene dvije skupine zahtjeva. Korisnički

zahtjevi (user requirements) su apstraktni opisi zahtjeva za korisnike, a sustavski

zahtjevi (system requirements) predstavljaju detaljni opis potrebne funkcionalnosti.

4. Vrednovanje zahtjeva (requirements validation) obuhvaća provjeru realističnosti,

konzistentnosti i kompletnosti zahtjeva. Cilj ovih aktivnosti je otklanjanje pogrešaka i

propusta u zahtjevima.

**6. Koji su osnovni sadržajni elementi RUP?**

RUP obuhvaća sljedeće četiri faze:   
1. Pokretanje (inception) Cilj ove faze je uspostavljanje poslovnog modela (bussines case). Treba identificirati sve vanjske entitete (ljude i sustave) i njihove interakcije sa sustavom. Na temelju ovih informacija procjenjuje se poslovni doprinos sustava. Ako je poslovni doprinos nedostatan projekt se može zaustaviti.   
2. Elaboracija (elaboration) Cilj ove faze je razvoj razumijevanja domene problema, uspostavljanje arhitekturnog okvira za sustav, razvoj projektnog plana, i identifikacija ključnih projektnih rizika. Na kraju ove faze očekuje se model zahtjeva (može biti prezentiran primjenom UML-a), opis arhitekture i plan za razvoj softvera.  
3. Konstrukcija (construction) Cilj ove faze je dizajn sustava, programiranje i testiranje. Dijelovi sustava se razvijaju paralelno i integriraju. Na kraju ove faze očekuje se djelatan softver i odgovarajuća dokumentacija, spremno za isporuku.  
4. Tranzicija (transition) Završna faza odnosi se na preuzimanje softvera od strane korisnika i djelovanje u stvarnoj okolini. Ova aktivnost je zanemarena u mnogima modelima softverskih procesa, ali je značajna i potencijalno problematična. Na kraju ove faze očekuje se dokumentirani softver koji djeluje korektno u radnoj okolini.

**7. Koje prakse temeljene na načelima agilnih metoda uključuje ekstremno programiranje?**

1. Inkrementalni razvoj podržan je nizom malih, čestih inačica sustava. Zahtjevi se

temelje na jednostavnim korisničkim pričama ili scenarijima. Na temelju tih scenarija

se odlučuje koje funkcionalnosti će se uključiti u inkrement sustava.

2. Korisničko sudjelovanje podržano je kontinuiranim angažmanom korisnika u

razvojnom timu. Korisnički predstavnici sudjeluju u razvoju i odgovorni su za

definiranje prihvatnih testova.

3. Ljudi (a ne procesi) su podržani programiranjem u paru. Podupire se kolektivno

vlasništvo koda i održivi razvojni proces koji ne zahtijeva dugotrajan rad.

4. Promjena se prihvaća kroz niz primopredaja inačica sustava korisniku, razvoj s

testiranjem (test-first), i kontinuirani razvoj novih funkcionalnosti.

5. Održavanje jednostavnosti je podržana neprekidnim preuređivanjem (refactoring)

kojim se poboljšava kvaliteta koda i primjenom jednostavnog dizajna koji ne

pretpostavlja buduće izmjene sustava.

**8. Koje su značajke programiranja u paru?**

1. Programiranje u paru podržava koncept kolektivnog vlasništva i odgovornosti za

sustav. Tim je odgovoran za rješavanje svih problema s kodom, a ne pojedinac.

2. Programiranje u paru djeluje kao neformalni postupak revizije jer svaku liniju koda

razmatra najmanje dvoje članova tima. Ovaj proces znatno je ekonomičniji u odnosu

na formalne postupke inspekcije i revizije koda.

3. Programiranje u paru podržava preuređivanje koda čime se softver poboljšava. Važna

je značajka da se preuređenje provodi odmah i to pomaže svim članovima tima. Zbog

toga su članovi tima potaknuti sudjelovati i doprinijeti preuređenju koda.

**9. Koje su ključne značajke ciklusa sprintova u metodi SCRUM?**

1. Sprintovi su fiksne duljine, obično 2-4 tjedna. Odgovaraju razvoju inačice sustava u

XP.

2. Početna točke za planiranje je popis poslova ne projektu koje je potrebno obaviti

(product backlog). Za vrijeme faze procjene u okviru sprinta, to se revidira, određuju

se prioriteti i rizici. Korisnik je uključen u proces i može iskazivati nove zahtjeve na

početku svakog sprinta.

3. Faza izbora uključuje projektni tim koji surađuje s korisnikom za izbor značajki i

funkcionalnosti koje će se razviti u sljedećem sprintu.

4. Nakon faze izbora tim se organizira za razvoj softvera. Održavaju se kratki dnevni

sastanci svih članova tima na kojima se prati napredak i ako je potrebno redefiniraju

prioriteti. U ovoj fazi članovi tima su izolirani od korisnika i organizacije, a sva

komunikacija s njima odvija se preko scrum mastera. Uloga scrum mastera je vezana

za zaštitu razvojnog tima od vanjskih utjecaja. Način djelovanja ovisi o timu i vrsti

probleme. Scrum ne propisuje specifične tehnike za pisanje zahtjeva ili razvoj.

5. Na kraju sprinta obavljeni posao se revidira i prezentira zainteresiranim stranama

(stakeholders). Nakon toga započinje sljedeći ciklus sprinta.

**10. Koje su teškoće u uvođenju agilnih metoda u velike organizacije?**

1. Voditelji projekata koji nemaju iskustva u agilnim metodama mogu biti neskloni

prihvaćanju rizika zbog uvođenja novih pristupa jer ne znaju na koji način će to

utjecati na njihove projekte.

2. Velike organizacije često imaju procese i standarde vezane za kvalitetu koji su po

naravi birokratski i nekompatibilni s agilnim metodama. Ponekad su takvi procesi

poduprti softverskim alatima pa se tada njihova primjena smatra obveznom za sve

projekte.

3. Agilne metode najučinkovitije se primjenjuju ako su angažirani članovi tima s visokom

razinom vještina i sposobnost. U velikim organizacijama je prisutan veći broj

zaposlenika s različitim razinama vještina i sposobnosti. Ljudi s nižom razinom

sposobnosti mogu biti nedovoljno učinkoviti kao članovi timova u agilnim procesima.

4. Moguć je otpor uvođenju agilnih metoda u velikoj organizaciji s dugom tradicijom

primjene konvencionalnih procesa.