

RIS 3.ciklus Q&A

1. Nabroji pristupe programiranju i ukratko ih opiši.

Pristup programiranju

Monolitni pristup (build and fix)

- dugotrajno *kodiranje*, a zatim niz ponavljanja oblika *provjera+ispravak*
- odgađa otkrivanje problema (pogrešaka u kodu i dizajnu)
- prosljeđuje probleme u primjenu i održavanje

Inkrementalni pristup (stupnjevito, postupno programiranje)

- niz ponavljanja oblika *kodiranje+provjera+ispravak*
- omogućuje raniju provjeru i izdvajanje pogrešaka (fault isolation)
- omogućuje raniju raspoloživost djelomičnih (nedovršenih) verzija
- omogućuje ravnomjerniju podjelu posla

2. Koje su prednosti i nedostaci refaktoriranja? Nabrojite i ukratko opišite reprezentativne predloške refaktoriranja.

Prednosti refaktoriranja

- sprječava narušavanje strukture programskog koda
 - ako se nakon svake promjene koda izazvane promjenom zahtjeva obavi refaktoriranje, tada struktura ostaje očuvana, što olakšava buduće promjene
- Povećanje razumljivosti i čitljivosti programskog koda
- Olakšano otkrivanje bugova
- Povećanje produktivnosti
 - promjene koda brže obaviti ako je kod dobro dizajniran, lak za razumijevanje i bez bugova – ušteda vremena je znatno veća od vremena utrošenog na refaktoriranje
- Smanjenje troškova održavanja pojednostavnjenjem dizajna

Nedostaci refaktoriranja

- Prejerana primjena smanjuje produktivnost i svodi se na puko refaktoriranje
- Ukoliko nije automatizirano, refaktoriranje može biti dugotrajno i mukotrpno
- Postojeći alati nisu dovoljno zreli i pouzdani – nepovjerenje programera

Reprezentativni predlošci refaktoriranja

Rename method

- Davanje novog, razumljivijeg imena metodi

Extract method

- Dio koda se izdvaja u posebnu metodu

Replace temp with query

- Zamjena varijable koja poprima vrijednost nekog izraza s pozivom metode

Move method/field

- Metoda/članska varijabla se iz jednog razreda prebacuje u drugi razred

Extract class – inline class

- Razred koji “radi puno toga” dijeli su više razreda – *extract class*
- obrnuto – *inline class*

Replace conditional with polymorphism

- Zamjena uvjeta (*switch*) koji ispituje tip objekta s polimorfizmom, tako što se originalna metoda učini apstraktnom i u podrazredima se primjeni *overriding*

Replace type code with state/strategy

- Postoji kod koji utječe na ponašanje a ne može se primijeniti *subclassing*

3. Nabrojite i opišite stadije, tj. faze testiranja.

3.1. Opišite testiranje jedinica i njegove vrste testiranja.

Testiranje jedinica

Testiranje jedinica (unit testing), pojedinačno testiranje

- najmanja jedinica mjere je razred !
- testovi primjenjivi na nadređeni razred primjenjivi su na iz njega izvedene razrede, u dijelovima koji nisu preopterećeni nasljeđivanjem
- svaki put kad se razred koristi u različitom kontekstu treba biti ponovno provjeren s obzirom na posebnosti (višeobliče)

» vrste testiranja

Funkcionalno (black-box testing)

- provjera se što cjelina radi, to jest da li zadovoljava zahtjeve
- probni slučajevi izvode se iz specifikacija
- provodi osoblje proizvođača ili korisnici
- osnovica plana testiranja: CRC kartice, dijagrami razreda, ugovori

Strukturalno (white-box, clear box testing)

- provjera kako cjelina radi
- probni slučajevi izvode se uvidom u programski kôd (inspekcija koda)
- provode programeri
- osnovica plana testiranja: specifikacije metoda

3.2. Opišite integracijsko testiranje i njegove vrste testiranja.

Integracijsko testiranje

Integracijska provjera (integration testing)

- ispitivanje komponenti koje integrirane čine cijeli sustav ili neki njegov dio
- razine: razredi koji tvore logičku cjelinu, sloj, paket, knjižnica
- ispitivanje provodi tim (analitičara i programera) za testiranje

» vrste testiranja

Testiranje korisničkog sučelja (User Interface Testing)

- provjerava se svaka funkcija sučelja
- osnovica: dizajn sučelja
- testiranje se radi prolaskom kroz svaku stavku izbornika sučelja

Testiranje slučajeva korištenja (Use-Case Testing)

- provjerava se svaki slučaj korištenja
- osnovica: slučajevi korištenja
- testiranje se radi prolaskom kroz svaki slučaj korištenja.
- često se kombinira s testom korisničkog sučelja jer UC ne testira sva sučelja

Integracijsko testiranje (nastavak)

Testiranje interakcije (Interaction Testing)

- testiranje svakog procesa korak po korak
- osnovica: dijagrami razreda, dijagrami slijeda, dijagrami komunikacije
- Cijeli sustav započinje kao skup okrajaka. Razredi se dodaju pojedinačno i rezultati se uspoređuju s očekivanim rezultatima iz probnih podataka.
 - Nakon što jedan razred prođe testove, dodaje se sljedeći razred i ponavljaju se testovi.
 - To se radi za svaki paket.
 - Kada svi paketi prođu sve testove, proces se ponavlja za integriranje paketa.

Testiranje sučelja sustava (System Interface Testing)

- testiranje razmjene podataka s drugim sustavima
- osnovica: dijagrami slučajeva korištenja
- Budući su prijenosi podataka između sustava često automatizirani, a korisnici ih izravno ne prate, važno je oblikovati testove koji će osigurati da se prijenosi obavljaju ispravno.

3.3. Opišite testiranje sustava i njegove vrste testiranja.

Testiranje sustava

Provjera sustava (System Testing)

- provjera rada sustava kao cjeline, kojom se osigurava da svi nezavisno razvijeni aplikacijski programi rade ispravno te sukladno specifikacijama

» vrste testiranja

Testiranje zahtjeva (Requirements Testing)

- testiranje jesu li zadovoljeni izvorni poslovni zahtjevi
- osnovica: dizajn sustava, testovi komponenti i integracijski testovi
- Osigurava da promjene tijekom integracije nisu dovele do novih pogrešaka.
- Testeri se pretvaraju da su nekompetentni korisnici i izvode neprikladne radnje da testiraju robustnost (npr. dodavanje praznih ili duplih zapisu).

Testiranje uporabivosti (Usability Testing)

- testiranje prikladnosti sustava za korištenje
- osnovica: dizajn sučelja i slučajevi korištenja
- Često ih radi analitičar koji ima iskustva u razumijevanju korisnika kao i u dobrom dizajnu sučelja.

Testiranje sustava (nastavak)

Testiranje sigurnosti (Security Testing)

- testiranje mogućnosti oporavka i neautoriziranog pristupa
- osnovica: dizajn infrastrukture
- Testiranje sigurnosti je složen zadatak kojeg obično radi analitičar infrastrukture. U ekstremnim slučajevima, provodi zasebna tvrtka.

Testiranje performansi (Performance Testing)

- ispitivanje sposobnosti izvođenja pod velikim opterećenjem
 - stress testing - velik broj interakcija (simulacijom pristupa)
 - load testing – velika količina podataka (npr. generatorima podataka)
- osnovica: prijedlog sustava, dizajn infrastrukture

Testiranje dokumentacije (Documentation Testing)

- testiranje ispravnosti dokumentacije
- osnovica: sustav pomoći, postupci, priručnici
- Analitičar provjerava svaku stavku dokumentacije kako bi osigurao da su primjeri u dokumentaciji i ponašanje sustava usklađeni.

3.4. Opišite test prihvatljivosti i njegove vrste testiranja.

Test prihvatljivosti

Provjera prihvatljivosti (Acceptance Testing)

- test sustava kojim se dokazuje da proizvod zadovoljava korisničke zahtjeve i potrebe organizacije te uvjete pod kojima ga je naručitelj spremjan preuzeti
- iscrpan i konačan test nad stvarnim podacima

Alfa-testiranje (Alpha Testing) - verifikacijsko

- probna uporaba koju provode korisnici kod izvođača
- simulacija stvarnog okruženja
- traženje pogrešaka i propusta

Beta-testiranje (Beta Testing) – validacijsko

- provode korisnici kod sebe, bez nazočnosti izvođača
- provjera u stvarnim uvjetima
 - performanse sustava
 - vršna opterećenja
 - provjera upotrebljivosti i lakoće uporabe
 - radne procedure
 - izrada rezervnih kopija i oporavak sustava

Nadzorni test (Audit Test) – provodi se opcionalno

- potvrda da je sustav gotov, ispravan i spremjan za primjenu
- provode nezavisne tvrtke ili odjeli za osiguranje kvalitete

4. Navedite i opišite 2 vrste dokumentacije.

Sistemska dokumentacija

Sistemska dokumentacija

- dokumentira razvoj i proizvode pojedinih faza
- namijenjena tehničkom osoblju
- potrebna za razumijevanje, izradu i održavanje sustava
- glavnina (pisane) dokumentacije nastaje tijekom analize i projektiranja sustava (!?)
- programska dokumentacija dijelom se može generirati

upravljanje projektom i konfiguracijom sustava

- plan razvoja, specifikacija zahtjeva i dizajna, ...

programska dokumentacija

- izvorni kôd, opis baze podataka, probni podaci i rezultati provjere, dnevnik promjena, programski priručnici

upute za rukovanje i održavanje (installation, operations manual)

- opis instalacijske procedure
- opis procedura za pokretanje/zaustavljanje (startup/shutdown)
- opis izrade rezervnih kopija i vraćanja podataka (backup, restore)
- opis postupka ponovnog pokretanja i oporavka (restart, recovery)

Korisnička dokumentacija

Korisnička dokumentacija

- pomoć korisnicima pri korištenju aplikacija
- mora biti prilagođena korisnicima različitog iskustva
- korisnički priručnik, upute za uporabu (user manual)
- materijali za poduku, upute za vježbu (training guide, tutorial)
- dinamička pomoć (online help)

Vrste korisničke dokumentacije s obzirom na strukturu

- referentni priručnik (reference document), najčešće sustav pomoći
 - kad korisnik treba naučiti kako obaviti neku funkciju
 - obično se čita nakon što je intuitivan pokušaj obrade bio neuspješan, stoga treba biti prikladno/pažljivo napisan
- priručnik procedure (procedure manual)
 - opisuje kako obaviti poslovne zadatke koji se sastoje od više funkcija ili koraka, npr. ispis mjesecnog izvješća, zaprimanje narudžbe
- vodič, lekcija (tutorial)
 - podučava kako koristiti glavne komponente sustava
 - elementi dokumentacije dulji i oblikovani tako da se čitaju u slijedu

5. Navedite i opišite na koja 3 načina se provodi konverzija sustava.

Način konverzije

- **Izravno uvođenje (direct conversion, abrupt cutover, cold turkey, big bang)**
 - početak rada novog sustava uz istovremeni prestanak rada starog sustava
 - u poslovnim sustavima provodi se na određeni dan, uobičajeno datum završetka poslovnog razdoblja, po mogućnosti na kraju tjedna
 - mogući problemi: pojava pogrešaka koje nisu bile uočene tijekom testiranja, nepredviđeno preopterećenje opreme u punom pogonu
 - nedostatak: neposredna izloženost korisnika pogreškama sustava

- **Paralelno uvođenje (parallel conversion)**
 - istovremeni rad starog i novog sustava tako dugo dok se ne pokaže da novi sustav ispravno radi i da su se korisnici navikli na novi način rada
 - bitno manje rizičan postupak u odnosu na izravno uvođenje
 - nedostatak: potreba za dvostrukom obradom istih podataka, u starom i u novom sustavu → otpor korisnika

Lokacija konverzije

- **Lokacija konverzije**
 - dijelovi organizacije smješteni na različitim zemljopisnim lokacijama
 - u nekim slučajevima lokacijom se smatraju različite organizacijske cjeline u istom kompleksu (npr. prodaja, otprema, nabava)

- **Probno uvođenje (pilot conversion)**
 - izravno/paralelno uvođenje sustava na jednoj lokaciji ili u jednoj radnoj cjelini
 - nakon što sustav uspješno prođe pilot test, uvodi se na ostalim lokacijama
 - prednost: dodatna razina provjere prije širenja po organizaciji, pa se problemi odražavaju samo na probnu instalaciju
 - nedostaci: dulje trajanje, razdoblje u kojem različiti dijelovi sustava koriste različite verzije sustava što otežava razmjenu podataka

- **Postupno uvođenje, fazno uvođenje (phased conversion)**
 - uvođenje slijedno po grupama lokacija
 - jedna skupina lokacija, pa druga, ...
 - karakteristike slične probnom, s tim da zahtijeva manje osoba

- **Istovremeno uvođenje (simultaneous conversion)**
 - istovremeno uvođenje na svim lokacijama – najčešće izravno
 - uklanja heterogenost, ali zahtijeva više ljudi

Modularnost konverzije

□ Uvođenje cijelog sustava (whole system conversion)

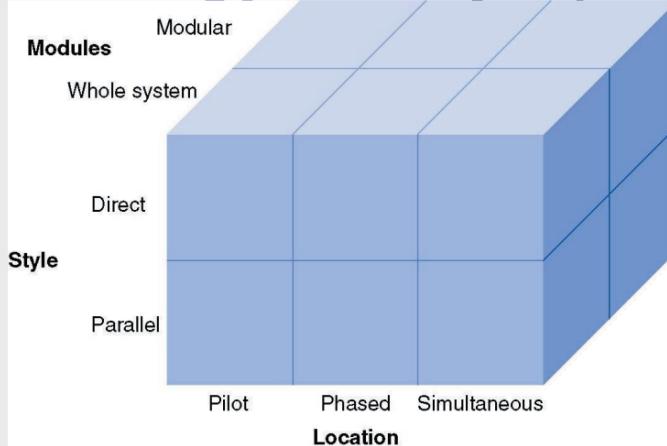
- čitav sustav instalira se odjednom – najčešći način
- u slučaju velikih sustava (npr. ERP, kao što su SAP ili PeopleSoft), može biti naporno usvajanje za korisnike

□ Modularno uvođenje (modular conversion, staged conversion)

- postupna zamjena starog sustava novim, uvođenjem po dijelovima
- izvedivo samo ako je moguć istovremeni rad oba (nekompletna) sustava
- problemi: potreba za spojnim programima, tj. premošćivanjem
 - svaki modul treba moći raditi i sa starim i s novim sustavom, ili
 - novi sustav mora moći učahuriti funkcionalnost starog
- prednost: postupni prijelaz, lakša poduka korisnika
- nedostatak: dulje trajanje

6. Popunite tablicu odabira strategije uvođenja u primjenu.

Odabir strategije uvođenja u primjenu



Uvođenje	Rizik	Trošak	Trajanje
Izravno	visok	nizak (ako uspije)	kratko (ako uspije)
Paralelno	nizak	visok	dugo
Probno	nizak	srednji	srednje
Fazno	srednji	srednji	dugo
Istovremeno	srednji	srednji	kratko
Cijeli sustav	visok	srednji	kratko
Modularno	srednji	visok	dugo

7. Navedite i opišite vrste održavanja sustava.

Vrste održavanja

Preventivno

- podrazumijeva zaštitu od mogućih problema
- redovita izrada sigurnosnih kopija (backup)
- obavlja se periodički (dnevno, tjedno, mjesечно)

Korektivno

- podrazumijeva popravak nakon što se problem pojavio
- vraćanje podataka iz sigurnosne kopije (restore)
- uklanjanje uzroka pogreške (ispravljanje programa)

Adaptivno

- prilagodba funkcionalnosti (načina posluživanja)
- prilagodba strukture (promjene strukture podataka)
- poboljšanje performansi (optimizacija programa)

Perfektivno

- nadgradnja sustava da bi se riješili novi problemi
- ugradnja novih mogućnosti (features)

8. Koje su karakteristike metodologija razvoja

Karakteristike metodologija

Kuhanje po kuharici (receptu) ne znači da će jelo biti dobro!

- Preporučene aktivnosti ne moraju uvijek biti prikladne, primjenjive ili potrebne
- Korisnik je nepredvidljiv i "prevrtljiv" - zahtjevi se mogu mijenjati u vremenu
- Izostiranje na propisanim procedurama vodi u zanemarivanje stvarnih problema
- Posljedica: formalno dobro napravljen, a neuspješan sustav

Nedostaci metodologija (općenito)

- Nedostatak standarda – velik broj varijanti postupaka i notacija
- (Ne)odmjerenost – (pre)komplikirane ili (pre)jednostavne
- Pokrivenost životnog ciklusa – rijetke podupiru sve faze životnog ciklusa
- Podržanost alatima – nepotpuna jer proizvođači alata nastoje približno podržati što veći broj notacija

Dodaci ili alternative metodologijama

- zdrav razum
- najbolje dokazano u praksi - "Best practices"
- prečice do rješenja problema temeljene na temelju sličnih iskustava - "Rules of thumb"

9. Popunite tablicu usporedbe tradicionalnih naspram agilnih metodologija

Tradicionalne naspram agilnih metodologija

	Proces	Prakse	Struktura projektne ekipe
Tradicionalni pristup	„Težak“: plan strogo definiran na početku – planski usmjeren (eng. plan driven); linearan i predvidiv	Nisu definirane. Gradi se postepeno, a isporučuje samo jednom.	Distribuirane ili kolocirane, uglavnom velike ekipe sa strogo definiranim ulogama
Moderno agilni pristup	„Lagan“: nema strogo definiranog plana, planiranje se odvija kroz cikluse, proces: nelinearan i prilagodljiv, inkrementalan	Sedam osnovnih: samoorganizirajuće ekipe, česta isporuka, planiranje učenja snažna komunikacija, testiranje svega, mjerjenje vrijednosti i „čišćenje puta“	Manje, kolocirane ekipe

Tradicionalne naspram agilnih metodologija (2)

	Dokumentacija	Tipovi programske podrške	Alati
Tradicionalni pristup	Dobro dokumentirane. Najprije dokumentirati, a onda razvijati prema definiranim dokumentima.	Bilo koji tip, ali s različitim rizikom. Inženjerski ili znanstveni. Veliki ili mali projekti.	Različiti alati za svaku fazu s kasnijom integracijom.
Moderno agilni pristup	Malo ili bez dokumentacije. Usmjereno na taktičko znanje – dijeljenje znanja između članova ekipe.	Poslovni sustavi, s promjenjivim zahtjevima. Manji projekti.	Integrirana razvojna okruženja.

10. Koje su glavne karakteristike URP-a (engl. RUP)?

Glavne karakteristike URP-a

Produktivnost ekipe

- baza znanja (u obliku web stranica) s detaljnim uputama o procesima, predlošcima dokumentacije te uputama za korištenje razvojnih alata
- osigurava da svi članovi dijele zajednički jezik, proces i pristup razvoju

Vizualno modeliranje

- naglašava razvoj i održavanje modela umjesto papirnate dokumentacije

Oslonac na UML

- tzv. industrijski standard

Podržanost alatima

- izrada i održavanje raznih artefakata (pretežno modela), vizualno modeliranje, programiranje, testiranje itd.
- alati pomažu razvoj ali i procese upravljanja izmjenama i konfiguracijom

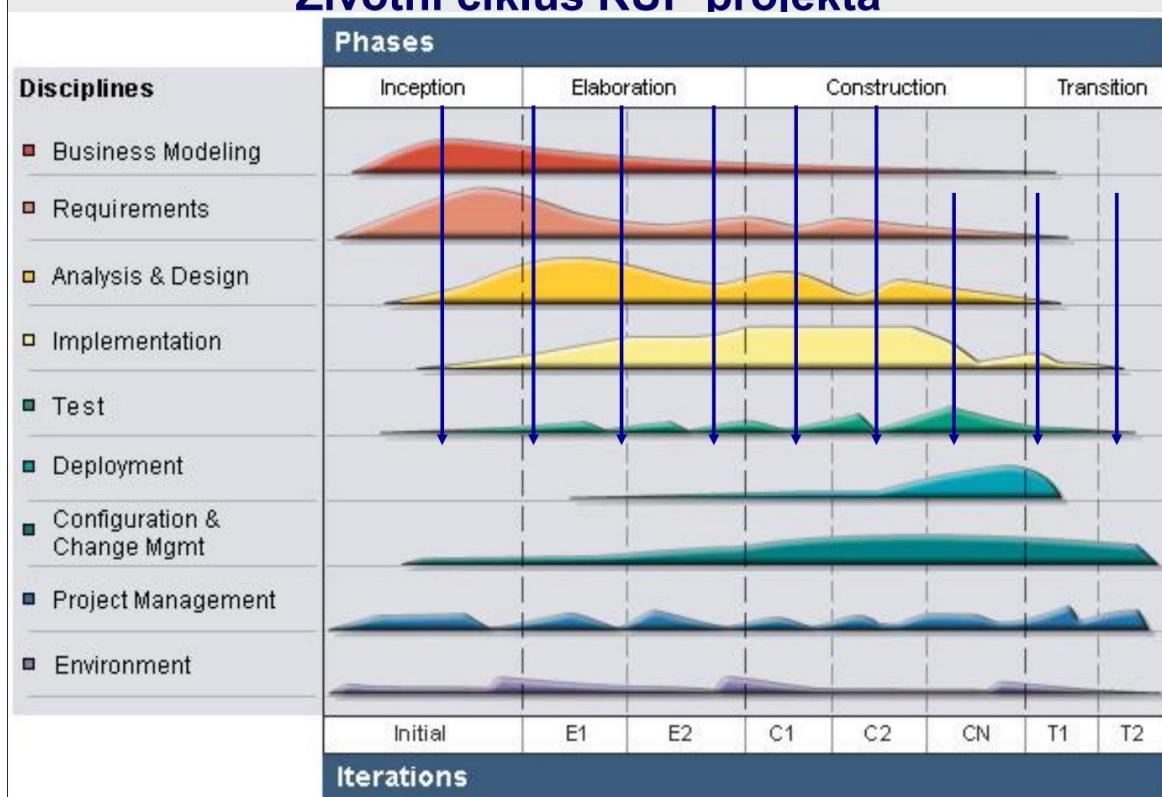
Iterativan i inkrementalan razvoj

Elastičnost i prilagodljivost

- prema veličini ekipa ili tipu projekta
- RUP sadrži niz "predložaka" razvojnih procesa (roadmaps) za različite modele razvoja i tipove projekata

11. Nadopunite na slici životni ciklus RUP projekta disciplinama i fazama razvoja. Opišite ih.

Životni ciklus RUP projekta



11.1. Navedite discipline RUP projekta i opišite ih.

Glavne discipline

- Poslovno modeliranje (Business Modeling)**
 - Uspostavlja poslovni kontekst sustava te oblik organizacije u kojoj sustav treba uvesti u primjenu.
 - Definiraju se ciljevi i okvirna funkcionalnost, poslovna pravila i sl.
- Requirements (Zahtjevi)**
 - Definira kako saznati i prikupiti želje zainteresiranih strana te ih pretvoriti u skup zahtjeva koji definiraju doseg sustava i potrebnu funkcionalnost.
- Analiza i dizajn (Analysis & Design)**
 - Definira pretvorbu zahtjeva u dizajn
 - Analiza usmjerena na logički pogled i funkcionalne zahtjeve
 - Dizajn usmjerjen na fizički pogled i nefunkcionalne zahtjeve
- Implementacija (Implementation)**
 - Kako razviti, organizirati, testirati i integrirati komponente
- Provjera (Test)**
 - Kako testirati i procijeniti kvalitetu rješenja
- Uvođenje u primjenu (Deployment)**
 - Aktivnosti potrebne da sustav bude dostupan svim krajnjim korisnicima

Podupiruće discipline

- Upravljanje konfiguracijom i promjenama (Configuration & Change Management)**
 - Disciplina koja opisuje kako kontrolirati i sinkronizirati evoluciju skupa komponenti i isporuka koje zajedno čine konačni sustav
- Upravljanje projektom (Project Management)**
 - Disciplina usredotočena na planiranje projekta, upravljanje rizicima, praćenje napretka i metriku
- Okolina (Environment)**
 - Organizira dijelove metodologije koji pružaju okruženje razvojnog timu, uključujući procese i alate

11.2. Navedite glavne faze razvoja RUP projekta i opišite ih.

Glavne faze razvoja - Počinjanje

□ Faza incepције (Inception phase) - Počinjanje

- Formuliranje opsega projekta
 - opis problemskog konteksta te najvažnijih zahtjeva i ograničenja
 - prikupljanje najvažnijih zahtjeva (10% detaljno)
 - preporuča se istaknuti i kritične scenarije korištenja (UC scenariji)
- Inicijalna procjena ukupnog troška, vremena i rizika
- Planiranje i priprema poslovnog slučaja
- Izrada prijedloga arhitekture
 - Cilj je demonstrirati izvedivost projekta putem nekog modela za simulaciju, inicijalnog prototipa i sl.
- Priprema okruženja za projekt
 - Procjena projekta i organizacije, odabir alata, razvojnih okruženja i procesa

Glavne faze razvoja - Elaboracija

□ Faza elaboracije (Elaboration phase) - Razrada

- Definiranje, validacija i zacrtavanje arhitekture
- Osiguranje da su arhitektura, zahtjevi i planovi stabilni, a rizici ublaženi
 - tako da se može pouzdano odrediti trošak i završetak projekta
- Prikupljanje detaljnih zahtjeva (80%)
- Ažuriranje vizije projekta
 - dobrom razumijevanjem kritičnih slučajeva korištenja koji su ujedno i nositelji najvećih rizika
- Izrada plana iteracija za fazu konstrukcije
- Dorada razvojnog procesa i uspostava razvojnog okruženja
 - uključujući proces, alate i podršku za automatizaciju
- Dorada arhitekture i odabir komponenti
 - Radi se procjena potencijalnih komponenti kako bi se mogla odrediti cijena i trajanje naredne faze

Glavne faze razvoja - Konstrukcija

□ Faza konstrukcije (Construction phase) - Izgradnja

- Upravljanje resursima, kontrola projekta i optimizacija procesa.
 - Paralelni razvoj nekoliko razvojnih timova s ciljem ubrzanja razvoja
- Završetak iterativnog i inkrementalnog razvoja konačnog proizvoda
 - Provjera prihvatljivosti
 - podrazumijeva dovršetak analize, dizajna, razvoja i testiranja
- Procjena razvijenih isporuka naspram definirane Vizije projekta
- Provjera da li su programska podrška, lokacije i korisnici spremni za beta isporuku.

Glavne faze razvoja - Tranzicija

□ Faza tranzicije (Transition phase) - Prijelaz

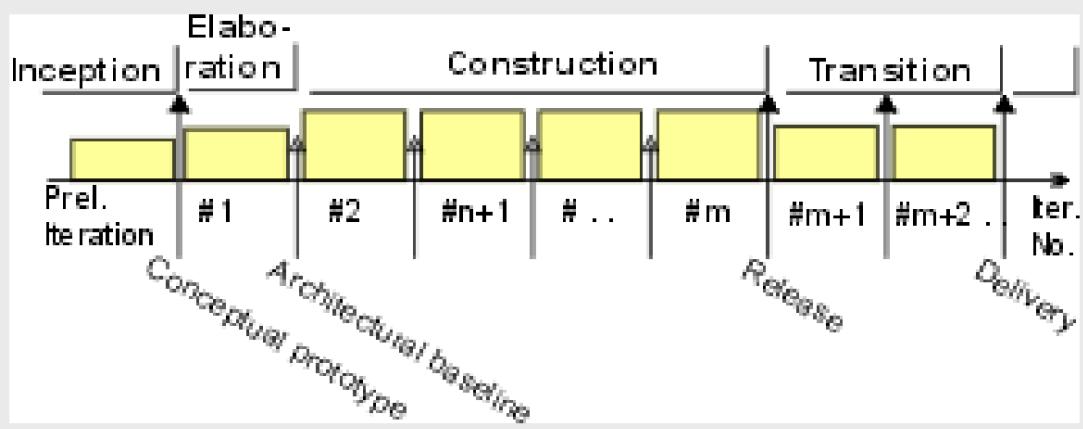
- Izvršavanje planova uvođenja u primjenu
- Dovršavanje korisničke dokumentacije i uputa
- Obuka krajnjih korisnika i održavatelja
- Testiranje programskog rješenja na lokaciji isporuke
- Izrada isporuke (release) konačnog programskog rješenja
- Prikupljanje povratne informacije od krajnjih korisnika
- Fino podešavanje rješenja (popravak manjih pogrešaka, poboljšanje performansi) na temelju povratne informacije
- Omogućavanje proizvoda dostupnim svim krajnjim korisnicima

12. Nabrojite i ukratko opišite strategije iterativne provedbe RUP projekta.

Strategije iterativne provedbe RUP projekta (1)

Inkrementalna strategija

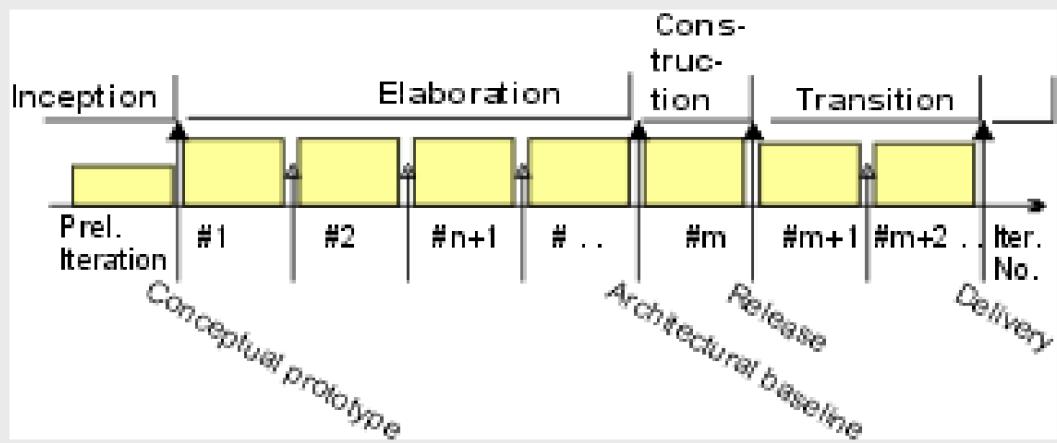
- poznata domena problema
- vrlo dobro poznati rizici
- iskusan projektni tim



Strategije iterativne provedbe RUP projekta (2)

Evolucijska strategija

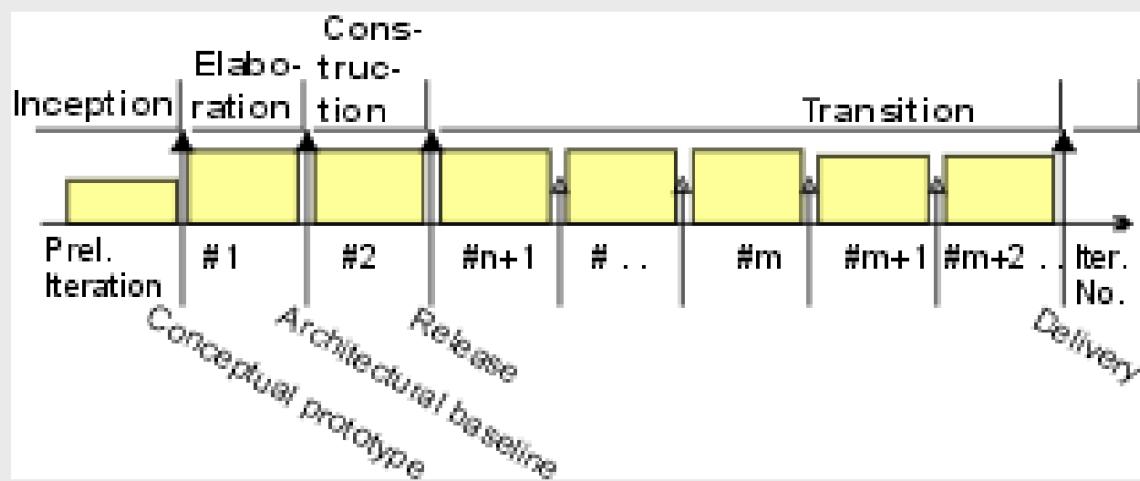
- nova ili nepoznata domena problema
- neiskusan projektni tim



Strategije iterativne provedbe RUP projekta (3)

□ Strategija inkrementalne isporuke

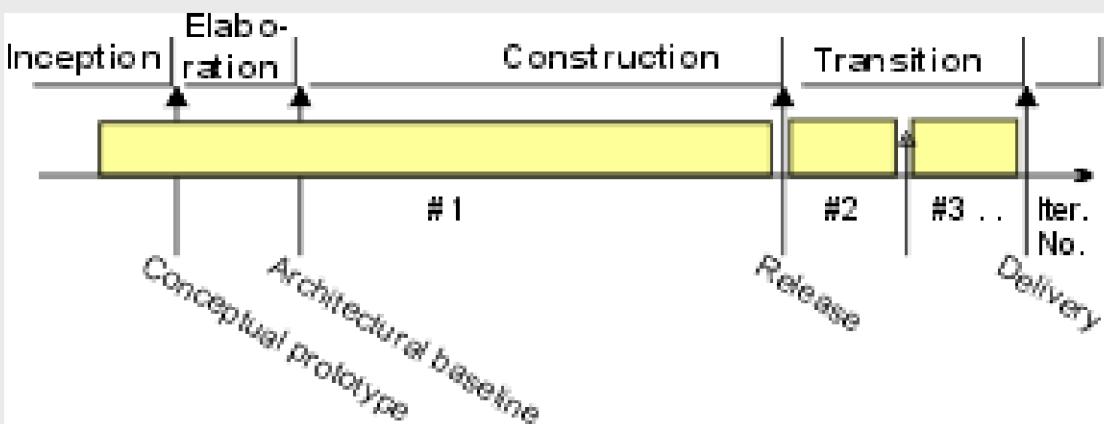
- poznata domena problema
- arhitektura i zahtjevi mogu biti stabilizirani rano u razvojnom ciklusu
- malo novog i nepoznatog u projektu
- iskusan projektni tim
- inkrementalne isporuke su vrlo važne i imaju visoku vrijednost za korisnika



Strategije iterativne provedbe RUP projekta (4)

】 Strategija "Velikog oblikovanja" (vodopadni model)

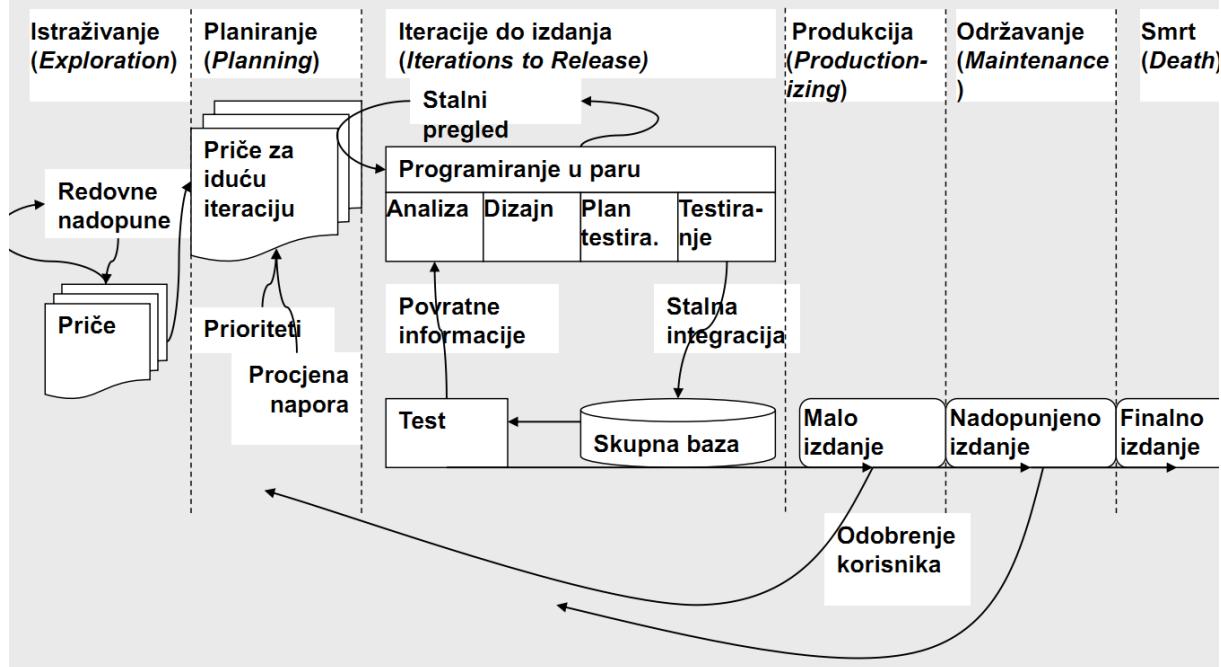
- dodaje se inkrement dobro definirane funkcionalnosti na vrlo stabilan proizvod
- nova funkcionalnost je vrlo dobro definirana i shvaćena
- tim posjeduje iskustvo kako u domeni tako i sa postojećim proizvodom



13. Nabrojite i opišite faze ekstremnog programiranja.

13.1. Životni ciklus ekstremnog programiranja

□ Životni ciklus



13.2. Faze...

□ Istraživanje

- Korisnici bilježe svoje priče na kartice
- Svaka kartica sadrži jednu mogućnost (feature) programa.
- Projektni tim se upoznaje s alatima, tehnologijom i postupcima projekta
- Radi se prototip sustava za testiranje tehnologije i varijanti arhitekture
- trajanje: nekoliko tjedana do nekoliko mjeseci

□ Planiranje

- Postavlja prioritete na korisničke priče (tj. svojstva programskog rješenja)
- Planira se doseg prvog malog izdanja i vrijeme za pojedinu karticu
- Zatim se određuje cjelokupni vremenski raspored
- Rok za izdavanje prvog malog izdanja obično je unutar dva mjeseca
- trajanje: nekoliko dana

Iteracije do izdanja

- Uključuje nekoliko iteracija sustava prije prvog izdanja
- Vremenski raspored iz faze planiranja se razlaže u više iteracija
- Prva iteracija stvara verziju koja obuhvaća cijelu arhitekturu ciljanog sustava
- Klijent određuje kartice koje će se koristiti pri svakoj narednoj iteraciji
- Testovi prihvatljivosti izvode na kraju svake iteracije
- Na kraju posljednje iteracije, sustav je spremna za produkciju
- trajanje pojedine iteracije: jedan do četiri tjedna

Producija

- Dodatno testiranje i provjera performansi sustava prije isporuke klijentu
- Razrješenje primjedbi te odlučivanje da li će se riješiti u tekucem izdanju
- Zakašnjele nove ideje i prijedlozi se dokumentiraju a implementacija odgađa
- trajanje pojedine iteracije: tri dana do najviše tjedan dana

Faze ekstremnog programiranja (3)

Održavanje

- Nakon što je prvo izdanje pušteno u produkciju
- treba istovremeno održavati softver u primjeni i proizvoditi nove verzije
- Zbog toga se brzina implementacije smanjuje
- Održavanje može zahtijevati nove članove tima i promjenu strukture tima

Faza smrti je blizu kada klijent nema više novih kartica s pričama

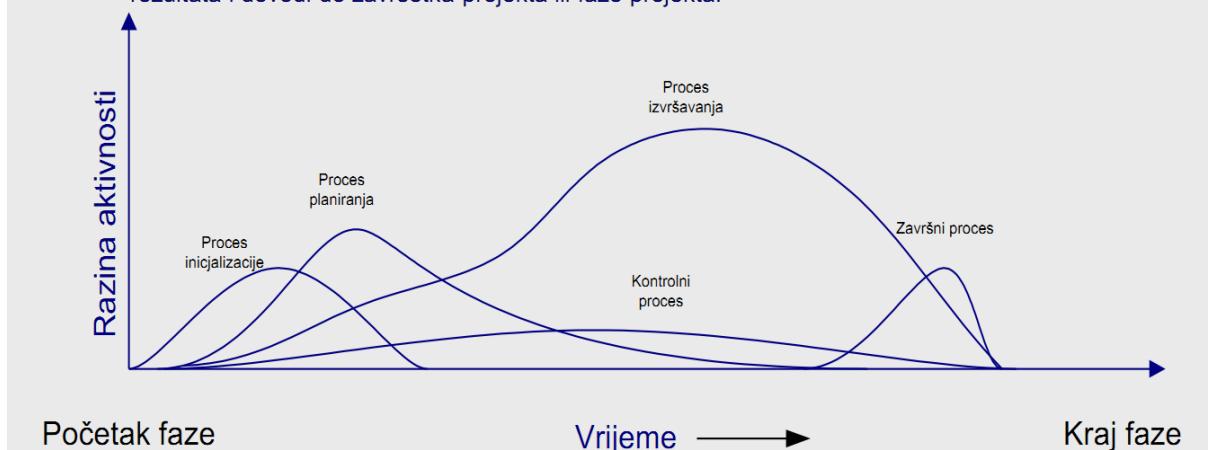
- Podrazumijeva se da sustav zadovoljava sve zahtjeve (npr. pouzdanost i stabilnost)
- Vrijeme u XP projektu da se konačno napiše sva korisnička dokumentacija budući da više nema promjena na arhitekturi, dizajnu i kodu sustava
- Smrt može nastupiti i kada sustav ne ispunjava sva korisnička očekivanja, ili ako postane preskup za daljnji razvoj

14. Opišite inženjerstvo prema naprijed i unatrag.

- Inženjerstvo prema naprijed (forward engineering)**
 - tradicionalni postupak kod kojeg s viših razina apstrakcije prelazimo na niže slojeve projektiranja i izvođenja
 - pojmu inženjerstvo se dodaje oznaka „naprijed“ samo da bismo ga kao tradicionalni pristup razlikovali od inženjerstva unatrag
- Inženjerstvo unatrag (reverse engineering) - obrnuto, reverzno, povratno**
 - kontekst
 - izvorni kod je dostupan, ali je dokumentacija nepotpuna, neažurna ili je nema
 - izvorni kod nije dostupan i cilj je doći do njega
 - glavne aktivnosti
 - prepoznavanje dijelova sustava i njihove povezanosti, s ciljem da se identificiraju komponente sustava i dobije potpuna specifikacija
 - oblikovanje sustava na višim slojevima apstrakcije iz nižih slojeva, npr. izrada dijagrama temeljem izvornog koda programa
 - ne podrazumijeva izmjene u postojećem sustavu, već se svodi na detaljno ispitivanje postojećeg sustava s ciljem stvaranja boljeg novog sustava
 - postupci - ispitivanje postojeće implementacije sustava, ponovno otkrivanje dizajna sustava, dešifriranje zahtjeva ugrađenih u sustav

15. Koje su grupe procesa projekta?

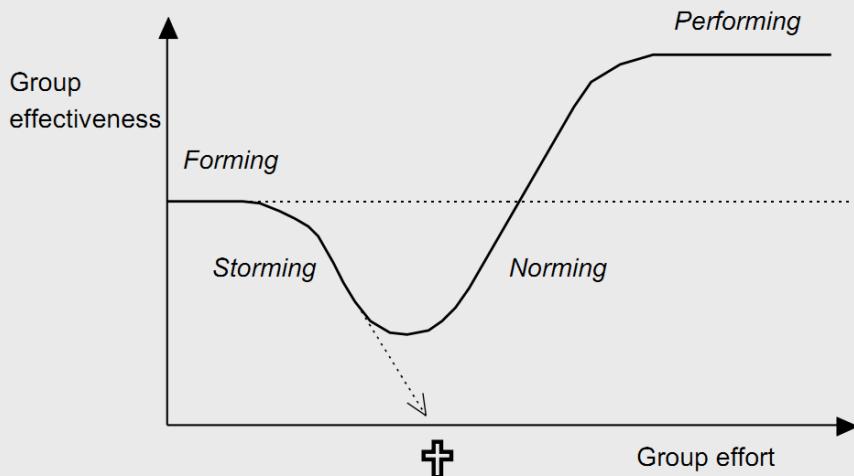
- Grupa procesa započimanja (Initiating Process Group) – definira i odobrava projekt ili fazu projekta.
- Grupa procesa planiranja (Planning Process Group) – definira i istaćava svrhu, planira smjer i akcije za postizanje cilja i dosega.
- Grupa izvršnih procesa (Executing Process Group) – koordinira ljudske i druge resurse u svrhu provedbe plana
- Grupa upravljačkih procesa (Monitoring and Controlling Group) – mjeri i prati napredak radi uočavanja odstupanja od plana s ciljem poduzimanja korektivnih akcija
- Grupa završnih procesa (Closing Process Group) – formalizira prihvatanje proizvoda, usluge ili rezultata i dovodi do završetka projekta ili faze projekta.



16. Navedite tok razvoja ekipe.

Razvoj ekipe (Forming, Storming, Norming, Performing)

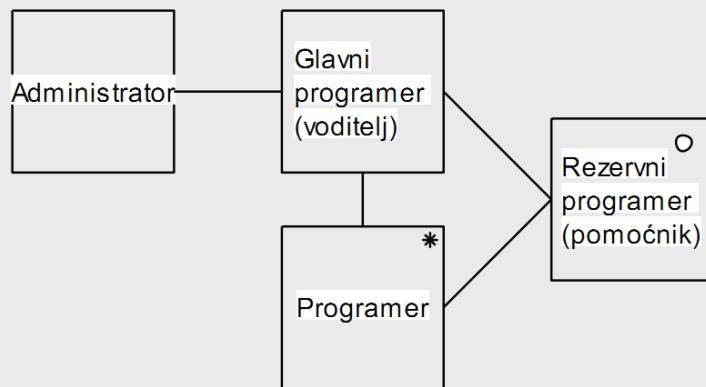
- Formiranje – ljubaznost, nesklonost iznošenju stavova, prepustanje vođenju
- "Jurišanje" – nesloga, sukob osobnosti, grupašenje/strančarstvo, pomanjkanje kvalitetne komunikacije, nesposobnost dogovaranja
- Normiranje – uviđanje dobrih strana zajedničkog rada, uvažavanje
- Predstavljanje, djelovanje – povezivanje u učinkovitu operativnu grupu



17. Navedite i opišite modele ekipa.

Ekipa s glavnim programerom (Chief Programmer Team) - klasična

- glavni programer (chief programmer) utjelovljuje znanje i odlučivanje ekipe
 - mora istovremeno biti dobar (vrhunski) programer i voditelj
 - u poboljšanoj (revidiranoj) organizaciji ima ulogu voditelja ekipe
- rezervni programer (backup programmer)
 - služi kao zamjena za nekog od mlađih (junior) programera
 - u poboljšanoj (revidiranoj) organizaciji ima ulogu pomoćnika voditelja



□ Moderna organizacija ekipe – (4GL ekipa, poslovna ekipa)

- voditelj projekta (project leader) – viši sistem analitičar
- suradnja s korisnikom (user liaison) – poslovni analitičar
- konceptualno i logičko oblikovanje – sistem analitičar
- isporuka sustava/aplikacija – poslovni analitičar
- nabava i pogon opreme – sistem inženjer za računala
- mrežni servisi – sistem inženjer za komunikacije
- programsko inženjerstvo – programer-analitičar
- izrada dokumentacije – urednik/pisac (editor / technical writer)
- potporno osoblje: administrativni koordinator, tehničari, činovnici

XP ekipa

- Razvojnik (developer)
 - kodiranje, pisanje testova, restrukturiranje programskog koda
- Klijent, korisnik
 - piše korisničke priče i testove prihvatljivosti
 - određuje redoslijed i prioritete implementacije
- Tester
 - pomoći korisniku pri izradi testova prihvatljivosti
 - izvođenje testova, objava rezultata testiranja, održavanje alata za testiranje
- „Pratilac“ (tracker)
 - prati procjene ekipe (npr. procjena utrošenog vremena) ocjenjuje njihovu točnost
 - prati napredak svake iteracije i procjenjuje uspješnost, predlaže promjene
- Trener
 - osoba zadužena za XP proces u cjelini, vodi ekipu kroz proces
- Savjetnik
 - vanjski član koji posjeduje specifično tehničko znanje koje je potrebno za projekt
- Upravitelj (Veliki šef)
 - donosi odluke, komunicira s članovima, prepoznaje teškoće, uvodi rješenja

□ Skunkworks tim

- grupa kreativaca odvojenih od utjecaja birokracije u organizaciji te prepuštena da po vlastitom nahođenju razvija produkt i/ili inovacije
- crna kutija – menadžment ne želi znati detalje posla, traži rezultate
- prednosti: sjećaj privrženosti projektu, produktivnost, motivacijski učinci
- nedostatci: nedovoljan uvid u napredak projekta

□ Tim orientiran na funkcionalnost (feature team)

- osoblje odgovorno za razvoj, kvalitetu, dokumentaciju i marketing organizirano je hijerarhijski prema funkciji (npr. hijerarhija razvojnika)
- timovi koji iz svake od funkcijskih grupa dobivaju jednog ili više članova, a svaki tim ima odgovornost za jednu od funkcionalnosti proizvoda
- balansirana matrična struktura

Spasilački tim

- tim usredotočen na rješenje specifičnog problema kod kojeg je potrebna hitna intervencija
- članovi - specijalisti za rad sa specifičnim alatima u određenom okruženju ili na posebnoj platformi
- primjer: ekipa koja intervenira prilikom pada telefonske centrale

Tim specijalaca (SWAT)

- Special Weapons and Tactics → Specialists With Advanced Tools
- prema modelu vojnih ili policijskih specijalnih jedinica
- članovi imaju veliko iskustvo u radu s određenim alatom ili paradigmom te im se prepušta rješavanje odgovarajućih problema
- članovi su uglavnom stalni i naviknuti raditi zajedno te svaki ima dobro definiranu ulogu

Tim profesionalnih atleta (*professional athletic team*)

- prema modelu sportskih ekipa
- članovi su jednakovražni ako ne i važniji od voditelja
- voditelj postoji da bi uklanjao prepreke i omogućio učinkovit rad
- članovi imaju strogo specijalizirane uloge i od eksperta za neko područje ne očekuje se da se bavi drugim područjem

Kazališni tim

- "režiser" prema viziji projekta dodjeljuje članovima zadatke i odgovornosti za pojedina područja
- članovi tima mogu oblikovati svoje uloge i svoje dijelove projekta prema vlastitom nahođenju, ali se njihove ideje moraju poklapati s vizijom
- uloge se pridjeljuju na temelju "audicije"
- nakon što su podijeljene, ne mogu se mijenjati

18. Popunite tablicu prikladnim ekipama za određene projekte.

	Rješavanje problema	Kreativnost	Taktičko izvršavanje
Dominantna značajka	Povjerenje	Autonomija	Jasnoća zadataka
Tipičan primjer	Korektivno održavanje	Razvoj novog produkta	Nadogradnja produkta
Naglasak u procesu	Fokusiranje na probleme	Istraživanje novih mogućnosti i alternativa	Dobro definirani zadaci s jasnim ulogama, te kriteriji uspjeha ili neuspjeha
Prikladni modeli razvoja	Kodiraj i popravi (<i>engl. code-and-fix</i>)	Spirala, evolucijsko prototipiranje ili isporuka, dizajn po rasporedu ili alatu	Vodopad, modificirani vodopad, spirala, dizajn po rasporedu, dizajn po alatu
Kriteriji za odabir članova	Inteligenca, suradnja, razmišljanje, osjetljivost, integritet	Kreativnost, neovisno razmišljanje, snažna motivacija, ustrajnost	Odanost, predanost, orijentacija na akcije, osjećaj za hitnost rješavanja problema
Prikladni modeli timova	4GL, SWAT, spasilački tim	4GL, CPT, skunkworks, kazališni tim, feature tim	4GL, CPT, SWAT, feature tim, tim profesionalnih atleta

19. Koje su karakteristike dobrih ekipa.

- Zajednička, inspirirajuća vizija ili cilj**
 - Cilj koji se postavlja mora biti inspirirajući, a posao izazovan (!?)
- Snažan identitet tima i osjećaj pripadnosti timu**
 - Uspjeh - samopouzdanje – povećani angažman – produktivnost – elitizam
- Struktura orijentirana na povećanje produktivnosti (zarade)**
 - Jasne uloge, dobra komunikacija, odluke na temelju činjenica, objektivan sustav nagrađivanja
- Kompetentni članovi**
 - Tehničke kompetencije (metodologije, platforme, programski jezici...), angažman i doprinos projektu, komunikacijske sposobnosti
- Predanost timu i organizaciji**
 - poticaji: vizija, izazov, identitet tima ili karizmatični vođa
 - predanost je moguća samo ako je osoba rasterećena problema u radnoj okolini (radna atmosfera, finansijska stabilnost)

- Međusobno povjerenje i međuzavisnost članova tima**
 - iskrenost, otvorenost, dosljednost i uvažavanje
- Učinkovita komunikacija**
 - tehnološka i verbalna
- Osnaživanje (*empowerment*) - osjećaj autonomije i ovlaštenja**
 - mogućnost poduzimanja akcija koje ekipa smatra potrebnim
- Mali broj članova**
 - pravilo 7 ± 2 – četiri osobe nisu dovoljne za stvaranje grupnog identiteta, deset i više je teško koordinirati
- Uživanje u poslu**
 - članovi dobrih timova vole biti produktivni, a ako rade posao koji vole, napravit će još više posla
 - može se povećati uz (ograničenu) zabavu i humor

20. Koje su karakteristike dobrog voditelja?

- Voditelj mora zastupati interes organizacije i interes članova tima**
- Voditelj mora biti podržan u provođenju svojih ovlasti jer inače gubi autoritet**
- Mogućnost nagrađivanja i kažnjavanja članova tima**
 - direktno ili indirektno preko viših razina upravljanja
- Objektivnost**
 - dobra procjena količine i kvalitete posla, shodno tome vrednovanje članova
 - jednaka mjerila za sve članove tima
- Tehnička kompetencija**
 - dovoljno znanje – razumijevanje problema i procjena napora
- Preuzimanje odgovornosti**
 - autoritet i odgovornost za donošenje “najbolje”, makar krive, odluke
- Brzo i efikasno rješavanje konflikata**
 - npr. rješavanje problema problematičnog člana