



# *Deo* **2**

## **Upoznavanje sa osnovama XP-a**

**Čas 4 Životni ciklus u razvoju softvera pomoću XP-a**

**Čas 5 Uloge u timu koji primenjuje XP**

**Čas 6 Praktične preporuke XP-a na delu**





# Čas 4.

## Životni ciklus u razvoju softvera pomoću XP-a

**NA PRETHODNOM ČASU SMO VIDELI KAKO XP PREVAZILAZI OPŠTE PROBLEME** vezane za razvoj softvera. Naučili ste kako da pomoći praktičnih preporuka koje daje XP rešavate ključne probleme. Ovaj čas se bavi ukupnim procesom rada u XP-u.

Na ovom času ćete naučiti:

- Kako izgleda životni ciklus projekta u XP-u
- Kako kupci definišu poslovnu vrednost i kako usmeravaju projekat
- Kako kupci i programer saraduju prilikom definisanja sistema
- Da podelite softver na isporuke i iteracije
- Kako se u toku iteracije vrši razvoj
- Kako da pomoću XP-a ugradite svoj softver
- Kako se u XP projektu vrši održavanje



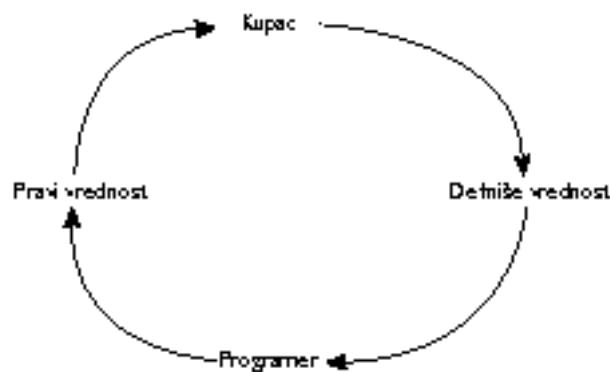
## Čas 4.

# Pregled životnog ciklusa projekta u XP-u

Osnovna premla XP-a je da kupac i programer treba da rade *zajedno* na proizvodnji softvera koji ima realnu vrednost. Kupac usmerava razvojni tim u tome kako da isporuče poslovnu vrednost. Kupac je aktivno uključen u razvoj proizvoda. Na prethodnom času ste naučili kako se u XP projektu prihvataju promene i neizvesnosti. Mi očekujemo da se dešavaju promene. Na slici 4.1 smo pokazali da je ciklus u XP-u proces stalnog definisanja i kreiranja vrednosti.

**Slika 4.1**

Kupac u XP-u  
definiše vrednost.  
Programer pravi i  
isporučuje vrednost.



Na izvesnom nivou se može reći da ovde nema ničeg novog. Sav razvoj se odvija oko vrednosti koju definiše kupac, a programer isporučuje tu vrednost. Razlika kod XP-a je u tome da se ova petlja izvršava vrlo brzo. Na svakih nekoliko minuta, sati ili dana tim pravi i isporučuje novu vrednost. Na taj način, preko sugestija i korekcija, kupac može da vodi ceo projekat. (Waterfall pristup dovodi do toga da kupac čeka nekoliko meseci pre nego što mu se isporuči neka vrednost.) U tabeli 4.1 je prikazan životni ciklus u XP-u:

**Tabela 4.1: Faze u XP-u**

Faza	Opis
Istraživanje	Početak projekta. Zahtevi korisnika na najvišem nivou, tehnički prototip.
Planiranje	Definisanje prioriteta u radu. Podela na isporuke i prvi plan.
Iteracije	Testiranje i razvoj sistema. Tu spada i planiranje iteracija u koji ma se dešava rad, ali podeljen do najnižeg nivoa. Krajnji korisnici mogu u ovoj fazi da rade i da utiču na fino podešavanje interfejsa. Na taj način se obezbeđuje upotrebljivost.

## Životni ciklus u razvoju softvera pomoću XP-a



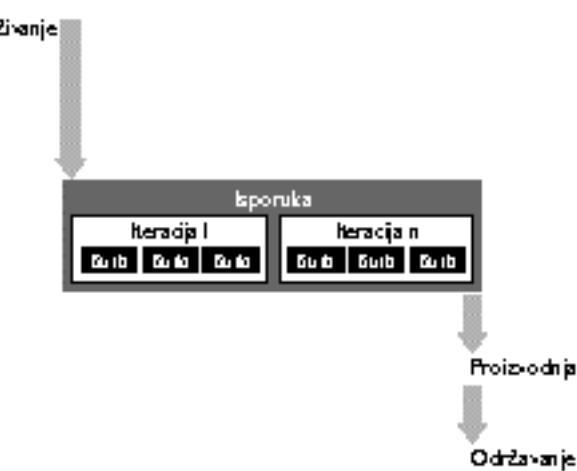
Tabela 4.1: Faze u XP-u

Faza	Opis
Uvođenje u proizvodnju	Ugradnja softvera u proizvodno okruženje kupca.
Održavanje	Stalno održavanje, "krpljenje" i proširivanje.

U najširem smislu XP projekti uključuju viziju kupca u isporuke, nakon čega se svaka isporuka deli na iteracije. Planiranje je evolucionaran proces, koji se odvija tokom celog životnog ciklusa projekta. Na slici 4.2 smo pokazali kako ovo radi.

Slika 4.2

XP isporuke se sastoje  
od iteracija



U toku iteracije postoji više kreiranja (rada), koja se dešavaju stalno u toku dana, kad god tim vrši integraciju novog koda. Broj kreiranja zavisi od tipa tehnologije koja se koristi i stila programiranja koji je tim prihvatio. Šta je osnovna razlika između isporuke i iteracije? Isporuka je ono što tim isporučuje kupcu, kao program koji nešto radi. Iteracija je interna stvar tima. Ona omogućava da kupac (koji se nalazi na licu mesta) i programeri izmere i prilagode svoj napredak.

### Novi termin

Kreiranje predstavlja proces u kome se komponente sistema integrišu u celinu, radi testiranja.

Na slici 4.3 smo ilustrovali iterativnu prirodu XP projekta.

Nakon istraživanja tim počinje ciklus planiranja i stalnih iteracija sve dok se proizvod kompletno ne osmisli i bude spremjan za proizvodnju. Isporuka je trenutak kada se softver prebacuje u "živo" ili radno okruženje. Proizvodnja je važna, pošto to znači da posao počinje da dobija svoju realnu vrednost, koja proističe iz Vašeg softvera. Druga strana ubacivanja ovakvog softvera u rad je da je cena vezana za neuspeh mnogo veća.



## Čas 4.

To objašnjava pažnju koju mnogi kupci posvećuju završnom testiranju, koje se odvija prilikom prijema programa.

**Slika 4.3**  
Životni ciklus  
projekta u XP-u



## Istraživanje zahteva korisnika

U fazi istraživanja se odvija otkrivanje vizije sistema i njegove misije. U ovoj fazi kupac ima obavezu da izloži svoju viziju. Vizija treba da bude iskazana u 20 do 30 reči i ona vodi do razvoja metafore sistema. Kupci i programeri u ovoj fazi rade zajedno. Istražuju se tehničke opcije, definišu zahtevi i kompletira lista *priča korisnika*.

### Novi termin

*Vizija je iskaz najvišeg nivoa koji opisuje cilj sistema. Na primer, svrha novog sistema je da prodaje knjige preko Interneta.*

### Novi termin

*Metafora sistema je način na koji tim vrši konceptualizaciju sistema i obično se piše u jeziku koji je relevantan za domen posla. Na primer, sistem se može shvatiti kao virtualna prodavnica knjiga u koju kupci (korisnici) mogu da dođu i potraže ili kupe knjige.*

### Novi termin

*Priča korisnika je alat koji omogućava izražavanje zahteva kupca. Priče korisnika piše kupac na jeziku koji nije tehnički. Priče korisnika se obično sastoje od nekoliko rečenica (3-4).*



## Pisanje korisničkih priča

U početnoj fazi, kupac razvija korisničke priče najvišeg nivoa, a programeri procenjuju vreme potrebno za implementaciju. Postoji implicitno slaganje da su procene napravljene u ovoj fazi grube i da će u toku napretka projekta biti fino prilagođene. U nekim slučajevima programeri detaljno razmatraju rešenja (provode vreme u izradi prototipa i tehničkim istraživanjima) da bi se dobila tačnija procena. Mogu se koristiti i specijalisti za tehnologiju ili konsultanti koji proveravaju ispravnost pristupa na najvišem nivou, a mogu da pomognu i prilikom provera. Koncept provere je jednostavan, ali snažan alat koji može da pomogne timu da odredi koji pristup razvoju treba da upotrebi. Ponekad se tehnički deo tima odvaja u parove, pri čemu svaki od parova istražuje konkretnu tehničku opciju.

### **Novi termin**

*Provera tehničkih rešenja je napor koji se ulaže na razvoj koda koji služi kao prototip. To nije samo vreme potrošeno na opšta tehnička istraživanja, to je istraživanje sa određenim ciljem.*

Na kraju izrade prototipa kod se odbacuje, a razvojni tim zna pravac kojim treba krenuti. Dužina ovog perioda se može meriti satima, ali su obično u pitanju dani. U toku napretka programeri mogu da naprave pojedine alate koji pomažu prilikom rešavanja tehničkih problema, ali je ovo pre izuzetak nego pravilo.



### **Napomena**

*Kod koji nastane u toku ove faze često se odbacuje, pošto kvalitet nije dovoljan da bi kod mogao da se koristi ponovo. U ovoj fazi programeri vreme troše na pisanje koda koji služi za istraživanje i nema pravu primenu.*

## Procena i otkrivanje

Faza istraživanja se uglavnom odnosi na otkrivanje. Kupci otkrivaju šta žele, a programeri traže mogući pristup i vrše procene. Dok programeri prave prototipove za različita rešenja, takođe, vrše procene i pretraživanja. To se radi na sledeći način: proceni se vreme potrebno za izradu prototipa, nakon čega se taj prototip napravi, a zatim se uporedi sa procenom. Na taj način u fazi istraživanja razvojni tim trenira svoje mogućnosti pravilne procene. Ovo je vrlo bitno kasnije u fazi planiranja iteracija.

### **IZRADA PROTOTIPNIH REŠENJA U STVARNOSTI**

Do potrebe za izradom prototipa sam došao prilikom izrade jednog projekta. Programirali smo magacin sa podacima (data warehouse) i bilo nam je porebno da ogromnu količinu podataka iz tekstualnih datoteka prebacimo u bazu podataka. Ja i partner proveli smo nekoliko minuta za tablom u pronalaženju rešenja. Prvo smo analizirali da li su tehnička rešenja realna, a nakon toga da na osnovu te analize, procenimo i vreme potrebno za obavljanje posla.



## Čas 4.

Uskoro smo zaključili da postoje dve opcije. Jedna je da se za učitavanje koriste alati same baze podataka, a druga opcija je da sami napravimo neki program. Uzeli smo fiksni vremenski period (manje od jednog sata) da brzo napravimo radni model koji bi u najmanju ruku mogao da odgovori na pitanje "da li ovo korisničko učitavanje radi".

Na kraju izrade prototipa znali smo u kom smeru treba da idemo sa razvojem. Pored toga, otkrili smo i da postoje okolnosti vezane za mrežu, koje će verovatno uticati na učitavanje.



### Napomena

*Faza istraživanja može da se upoređi sa razvojem poslovnih slučajeva ili sa definisanjem projekata. U XP-u programeri programiraju i tokom izrade prototipa. Kod klasičnog projekta to nije slučaj.*

Dužina faze istraživanja zavisi od prirode projekta i tehnologija koje su u igri. Ako se koriste poznati alati i tehnike, onda je i vreme potrebno za izradu prototipa manje. Ukupna dužina ove faze može biti nekoliko nedelja, pa do nekoliko meseci, u zavisnosti od domena i zahteva.

Izlaz iz istraživanja treba da bude metafora sistema i prva lista sa pričama korisnika. Sada, tim u grubim crtama zna šta se traži, kako će to da uradi i u kom pravcu treba da se kreću. To će u toku rada verovatno promeniti, što i jeste razlog što smo izabrali XP.

## Kreiranje plana projekta

Planiranje je kratka faza u kojoj kupci i programeri treba da se slože o osbinama prve isporuke. Osobine (priče korisnika) se isporučuju na način koji ima i poslovno i tehničko značenje. U tabeli 4.2 smo dali listu odgovornosti kupaca i programera u toku faze planiranja.

**Tabela 4.2: Odgovornosti u toku faze planiranja**

Uloga	Odgovornosti
Kupac	Definiše priče korisnika. Odlučuje koju poslovnu vrednost ima priča. Odlučuje koje se priče rade u prvoj isporuci.
Programer	Procenjuje koliko mu vremena treba za ispunjenje zahteva iz svake priče. Upozorava kupca na tehničke rizike. Meri napredak razvojnog tima.

## Životni ciklus u razvoju softvera pomoću XP-a



### Napomena

Tabela 4.2 je preuzeta i adaptirana iz knjige *Planning Extreme Programming* (2001), autora Kent Becka i Martin Fowlera.

Planiranje se sastoji u davanju i uzimanju koje se odvija između kupca i programera. Ovde postoji ravnoteža između tehničke izvodljivosti i poslovne vrednosti. Programeri mogu da naprave sopstvene priče korisnika "nulte poslovne vrednosti" za infrastrukturu koja podržava osnovni sistem. Primer za ovo može biti ustanovljavanje testova i okruženja u kojima se radi.

Praktična preporuka iz XP-a, ŠIgra planiranja, je osnovni alat koji se koristi za planiranje. Igra planiranja obuhvata i kupce i programere. Radi se preko priča korisnika, procena i definisanja prioriteta.



### Napomena

To što se praktična preporuka iz XP-a, koja se odnosi na planiranje, zove igra planiranja može dovesti do problema. Da li je XP zabava i igra? Neke velike konsultantske firme zaziru od upotrebe nekih termina i menjaju ih prema svojim konvencijama. Ovo ćemo detaljnije objasniti na času 18, "Implementacija XP-a u Vašoj organizaciji".

Igra planiranja je neformalan i pragmatičan događaj. Zahtevi korisnika se pišu na karticama koje se koriste i o kojima se diskutuje. XP preporučuje upotrebu indeksnih kartica kako za priče korisnika, tako i za planiranje. Upotreba ovih kartica nije obavezna, tako da tim može da koristi i neki drugi alat za saradnju.



### Napomena

Neki korisnici XP-a priče korisnika pišu u programu za obradu teksta, nakon čega ih stampaju. Problem kod ovakvog načina je da u toku igre planiranja treba rukovati različitim listovima papira.

Izlaz iz ove faze je isporuka, a ne detaljna struktura rada. Kupci i programeri sada shvataju šta treba da se nađe u isporuci. Naredni korak je planiranje pojedinačnih iteracija.

## Podela isporuka na iteracije

Isporuka se deli na određeni broj iteracija koje obično traju od jedne do četiri nedelje. Inicijalna iteracija se fokusira na komponente arhitekture koje su potrebne da bi se postavio osnovni sistem. Ovo se može nazvati i iteracija "nulta poslovna vrednost" pošto su priče korisnika verovatno više tehničke i ne donose direktnе koristi za kupca.

Naredne iteracije treba da isporučuju poslovnu vrednost, koja vremenom raste. Kupac bira priče po želji, a programeri ih realizuju. Pristup tipa "kupac može da promeni svoje mišljenje" za programere može da bude frustrirajući. Jedina uteha je da je vrme iteracije



## Čas 4.

fiksirano. Fiksiranje vremena znači da se ne može menjati domen, već samo priča korisnika na kojoj se radi. U toku iteracije kupac ne može da menja svoju priču. Poenta je u tome da je dužina ciklusa toliko kratka da su rizik ili mogućnost promene vrlo mali.

### **Novi termin**

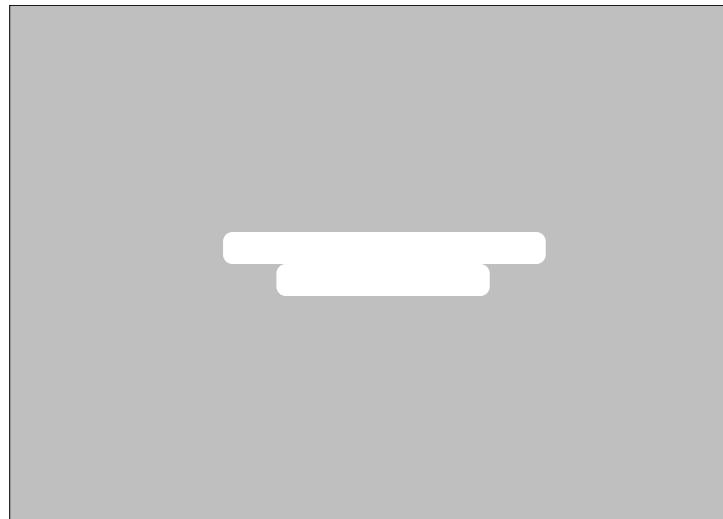
*Fiksiranje vremena (time boxing) je tehnika planiranja i kontrole projekata koji se odnose na razvoj softvera. Svaki vremenski okvir ima fiksnu dužinu, obično od jedne do četiri nedelje. Osnovni princip kod fiksiranja vremena je da se datum isporuke ne može promeniti. U zadatom terminu se mora isporučiti neki projekat.*

## Proces iteracije

Iteracija je mesto gde se odvija pravi rad. Na početku iteracije se održava sastanak na kome tim deli posao na programske zadatke. Kupac iz celog plana bira određenu priču. Prioritet se daje onima koje donose najveću poslovnu vrednost. Ako prethodna iteracija nije prihvaćena, to se ovde otkriva. Na slici 4.4 smo prikazali proces u okviru jedne iteracije.

**Slika 4.4**

*Ciklus jedne iteracije  
u XP-u*



Kupac može u toku rada da radi proveru upotrebljivosti. Ovo je bitno u slučajevima kada softverski proizvod treba da se "sretne" sa potrošačima. U slučaju javnog web sajta interfejs mora da zadovolji različite korisnike, koji imaju različite nivoe znanja i iskustva. Ako se provera upotrebljivosti ostavi za kraj isporuke, to nije tako efikasno kao provera u toku razvoja. Nema nekih utvrđenih pravila o tome kako upravljati razvojem interfejsa, dok se radi u malim iteracijama. Možda ćete zaključiti da je opravdano da sve vreme imate na raspolaganju grupu korisnika. Morate da obratite pažnju na to da ti korisnici brzo postaju iskusni, tako da više ne mogu da se tretiraju kao obični korisnici.



Korisničke priče su generalno suviše visokog nivoa, tako da se moraju razložiti na programerske zadatke, koji tu priču podržavaju. Programeri rade zajedno da bi pronašli zadatke koje treba da izvrše. Zadaci se zapisuju nazad u okviru priče, ili na posebnim karticama sa zadacima. Može se koristiti i neki drugi mehanizam za praćenje. Na času 10, "Prikupljanje zahteva od korisnika putem priča korisnika", ćemo dati neke primere priča.

Tipična dužina zadatka je jedan do tri dana idealnog programiranja. Idealan dan programiranja predstavlja vreme potrebno za rad, bez ikakvih poremećaja. Dugački zadaci treba da se podele na manje jedinice.



### Napomena

Tehnike merenja koje se koriste za procenu rada i prolaza objasnićemo u narednim poglavljima. Postoji više načina za merenje truda. Važno je da metrika koju koristite može da se prati i ponavlja.

Programeri se prijavljuju za odredene zadatke, a onda procenjuju koliko je potrebno za izvršenje njihovog zadatka. Najbolje je da zadatak obavlja programer koji je vršio *procenu*.

## Merenje prolaza

XP meri prolaz (količinu posla koji je tim obavio u toku određenog vremena) pomoću *brzine*. Realnost je da prva procena, za nultu iteraciju, bude netačna. Razlog je što je tim nov. Nisu jasno definisani zahtevi, a i alati koji se koriste još nisu potpuno ispitani. I kupac i programeri treba da znaju da je to početak. Oni treba da prate stvarni i procenjeni izlaz, tako da kasnije mogu da daju tačnije procene.

U toku razvoja, tim radi u parovima na zadacima koji su vezani za priče korisnika. Programerski par markira svoj zadatak kao završen. Ovo se radi na zidu sa centralnim planom. Na času 13, "Upotreba razvojnih alata za XP", možete naći primer ovakvog planiranja. Svakodnevni susreti se koriste kao forumi na kojima se brzo izlažu potrebne stvari i tu se sagledava status ukupnog napretka. Na ovim sastancima mogu da učestvuju i drugi, ne samo kupac i programeri. Oni brzo treba da shvate u kojoj fazi se projekat nalazi i da vide koliko dobro tim radi.

## Razvoj i testiranje

Programeri započinju rad na novom zadatku time što prvo pišu test. Taj test se dodaje u zbir svih testova. Programiranje se nastavlja u parovima. Svaki član para povremeno "seda" za tastaturu. Nakon završetka klase ili komponenti one se prebacuju na mašinu za integraciju. Mašina za integraciju je radna stanica koja se u potpunosti koristi za integraciju softvera. U toku iteracije, programeri stalno vrše integraciju. Ova integracija je podržana preko automatizovanih okruženja.

Na kraju iteracije, kupac izvršava test za prihvatanje, koji je pre toga napisan. Ako se neka korisnička priča ne prihvati, to se ispravlja u narednoj iteraciji.



## Čas 4.

# Puštanje softvera u rad u realnim uslovima

Na kraju isporuke, proizvod se proverava i odobrava se njegovo postavljanje u radno okruženje kupca. Ovo je trenutak kada eksperti za upotrebljivost treba da prikupe korisnike koji će izvršiti testiranje napravljenog. Testiranje sa realnim korisnicima garantuje da interfejs pomaže korisnicima u izvršenju njihovih zadatka. Sve što smanjuje produktivnost korisnika mora sada da se reši. To znači da isporuka još uvek nije kompletna.

Ova faza može da obuhvati i podešavanje sistema i aplikacije, u zavisnosti od ciljnog okruženja. Podešavanje se može izvesti putem isceniranog okruženja, prepostavljajući da to okruženje odražava izgled realnog okruženja. (Kupci obično zabranjuju da programer pristupi proizvodnom kodu.) Cilj nije da se nastavi funkcionalna evolucija, već da se izvrši stabilizacija sistema.

### Novi termin

*Iscenirano okruženje je korisničko okruženje koje daje sliku realnog okruženja. Ovakvo okruženje se koristi za završne testove pre konačne isporuke i za provjeru performansi. U razvoju softvera, obično, postoji sledeća okruženja: razvojno, za testiranje, inscenirano i proizvodno. Zanemarivanje podvojenosti ovih okruženja može da dovede do haosa u vreme ugradnje i održavanja softvera.*

Teoretski, bilo koji sistem koji je napravljen pomoću XP metodologije može se postaviti u proizvodnju čim se završi prva iteracija. (Prva iteracija bi mogla biti potpuno tehnička faza, u kojoj tim uspostavlja osnovni sistem.) Kupac bira iteracije koje se ugrađuju u zavisnosti od poslovnih vrednosti i spoljašnjih faktora, kao što je migracija podataka ili integracija sa postojećim sistemima. Vreme ugradnje će verovatno zavisiti od spoljašnjih ograničenja, a ne od mogućnosti tima da dovoljno brzo napravi kvalitetan softver.

Pristup za ugradnju softvera zahteva izvestan stepen planiranja i upravljanja rizicima. Razlog je što proizvodni tim sada ulazi u radno okruženje kupca. Prilikom postavljanja sistema u proizvodne uslove, morate biti svesni da će osoblje koje tamo radi zahtevati vodič za ugradnju i ostalu prateću dokumentaciju. Svaka dokumentacija koju razvojni tim bude pisao treba da bude u skladu sa pristupom koji se koristi u XP-u, što znači da dokumentacije treba da bude dovoljno, ali ne previše.

Da li će se kod ugradnje koristiti pristup velikog praska (big bang) ili fazna migracija, treba da odluče učesnici procesa. To su programeri, kupci, oni koji sa programom rade i krajnji korisnici. U tabeli 4.3 smo uporedili različite pristupe:

**Tabela 4.3: Poređenje različitih pristupa kod ugradnje softvera**

Pristup	Opis
Veliki prasak (big bang)	Jedno postavljanje posle završetka sistema. Ovaj sistem se koristi kada se menja postojeći sistem i kada je zadržavanje prelaznog perioda suviše složeno ili skupo. Ugradnja je jednostavna, ali su rizici veliki i greške mogu biti kritične za uspeh.

## Životni ciklus u razvoju softvera pomoću XP-a



Tabela 4.3: Nastavak

Pristup	Opis
Fazni	Novi sistem je razbijen na delove sa određenim funkcionalnostima. Ovaj pristup je dobar kada sistem može da se logički podeli na oblasti. Podela sistema u toku produženog perioda može da smanji rizik, ali se tako sistem izlaže promenama na duži rok.
Paralelni	U toku izvesnog perioda zajedno rade i novi i stari sistem. Ovakav pristup smanjuje rizik, ali postoji visoka cena koja se odnosi na održavanje i unos podataka na dva načina, kao i na dvostruko kreiranje odgovarajućih izveštaja.
Zeleno polje	Sistem se postavlja u okruženje u kome ne postoji nikakav sistem. Ovo bi moglo da bude prilikom početka novog posla, ili nove poslovne linije kojoj teba softverska podrška. To je idealan scenario za ugradnju pošto su rizici mali, a integracija sa drugim sistemima minimalna.

### **Novi termin**

*Veliki prasak je ugradnja sistema u jednom trenutku.*

## Održavanje sistema nakon isporuke

Nakon ugradnje sistem ulazi u fazu stalnog održavanja. Sa XP-om stalno imate razvoj, refaktorisanje i fino podešavanje sistema. Mogućnost “krpljenja” ili ažuriranja sistema se poboljšava definisanjem automatizovanih platformi za testiranje. To znači da slobodno možete da menjate kod, ako osetite da je to potrebno.

Sada, kada radite sa proizvodnim sistemom ugradnja zahteva više pažnje. Morate da napravite strategiju za prebacivanje podataka i kasnije “krpljenje” sistema. Može se pojaviti i uticaj na razvojni tim, ako stalno budu morali da dele svoje vreme između rada na novim stvarima i popravci postojećih. Ovo je životna realnost, tako da umesto da zanemarite ovo stanje, koje nije idealno, treba da zapišete i izmerite novi prolaz tima. U skladu sa tim treba da se promene i procene za tekuću ili nove iteracije.

## Zaključak

Na ovom času ste naučili kako se u XP-u vrši planiranje, ugradnja i popravka. Projekat počinje sa periodom uzajamnog otkrivanja i istraživanja koja obavljaju programeri i kupac. Jasno je da kupac definiše i opisuje poslovnu vrednost (šta je važno), dok programeri procenjuju i kompletiraju posao koji je neophodan da bi se ta vrednost isporučila. Projekat se deli na kratke periode aktivnosti, iteracije, u kojima programeri



## Čas 4.

pišu testove i kod. Takođe, tu se može izvršiti i testiranje upotrebljivosti. Životni ciklus u XP-u je vrlo elastičan i može se menjati, tako da kupac može da promeni smer, ako je to potrebno.

Na narednom času ćete naučiti koje su uloge u XP projektu, kao i koje odgovornosti postoje u toku životnog ciklusa.

## Pitanja i odgovori

**P Gde u XP-u kupac daje potvrdu da je sve u redu?**

**O** XP ne koristi formalnu potvrdu da su zahtevi ispunjeni, pošto on zahteva da kupac i programeri rade zajedno na tim zahtevima u toku planiranja. Kupac proverava i potvrđuje izlaz koji je tim dobio kod završnog testiranja.

**P Mi već koristimo pristup koji je različit od XP-a. Možemo li se prebaciti na XP.**

**O** Da. Lakoća prelaska zavisi od različitih faktora, uključujući trenutni status projekta, alate koji se koriste, veze sa kupcima i tip projekta. Promena metodologije projekta može da dovede projekat u poteškoće ili da oduzme izvesno vreme, koje je potrebno za učenje, čime se gubi ono što ste smatrali da ćete dobiti prelaskom na XP.

**P Gde je faza dizajna u XP-u?**

**O** Dizajn je stalan proces koji se ne dešava odjednom. Svi programeri su istovremeno i dizajneri u toku napretka projekta.

**P Da li je kupac jedna osoba?**

**O** Ne uvek. Pominjanje jedne osobe predstavlja pojednostavljenje koje se ovde koristi da biste shvatili životni ciklus. Kupca najčešće predstavlja grupa ljudi, odgovornih za pojedine oblasti.

**P Do kojeg nivoa XP programeri treba da se bave interfejsom sa krajnjim korisnicima, prilikom definisanja upotrebljivosti i stepena prihvatanja?**

**O** Razvojni tim treba da komunicira i uči od krajnjih korisnika prilikom rada na pričama korisnika. Poseban izazov predstavlja ubacivanje korporacijskog duha na web sajt. Ne možete da očekujete da odeljenje za marketing razvija svoj grafički interfejs u toku razvoja programa. Možda bi to trebalo uraditi pre nego što programeri počnu da rade. Nema razloga zašto tim ne bi mogao da koristi tehnike planiranja iz XP-a i da unapred napravi grafički dizajn. Kao i kod web integracije, verovatno na početka rada postoji vizuelni dizajn i struktura strana. Neki elementi dizajna se lako prave u PhotoShopu, ali nisu tako jednostavni za kodiranje.



## **Radionica**

Ova radionica testira da li ste shvatili sve koncepte koje smo razmatrali na ovom času. Pre nego što predete na sledeću lekciju, trebalo bi da shvatite i naučite odgovore na postavljena pitanja, jer će Vam to biti od velike koristi.

## **Kviz**

1. Koju praktičnu preporuku XP koristi za omogućavanje planiranja projekta?
2. Zašto u projektu postoji iteacija "nulta poslovna vrednost"?
3. Kako programeri rade sa pričama korisnika?
  - a. Potreban rad procenjuje tehničko osoblje koje određuje ko šta radi.
  - b. Svaki par, na sastanku na kojem se planira iteracija, dobija svoj deo posla.
  - c. Priče korisnika se dele na zadatke. Programeri procenjuju zadatke i na njima rade u parovima.
  - d. Menadžer projekta prati prolaz i dodeljuje deo posla na bazi performansi programera.
4. Zašto se meri prolaz razvojnog tima?
  - a. Programeri ove informacije koriste za naplatu.
  - b. Moramo da shvatimo koliko se priča korisnika može završiti u toku jedne iteracije.
  - c. Informacije se koriste za fino podešavanje budućih procena.
  - c. Kupci koriste ovaj prolaz da izmere stepen završenosti projekta.

## **Odgovori**

1. Igra planiranja
2. Da bi se razvojnom timu omogućilo da napravi osnovnu infrastrukturu
3. c.
4. b. i c.

## **Aktivnosti**

1. Uporedite stavove kupaca i programera u XP-u i kod pristupa Waterfall.
2. Razmotrite važnost ljudskog aspekta i apseksa veza kod XP-a. Da li su ovi aspekti u XP-u važniji nego kod drugih metodologija i zašto?