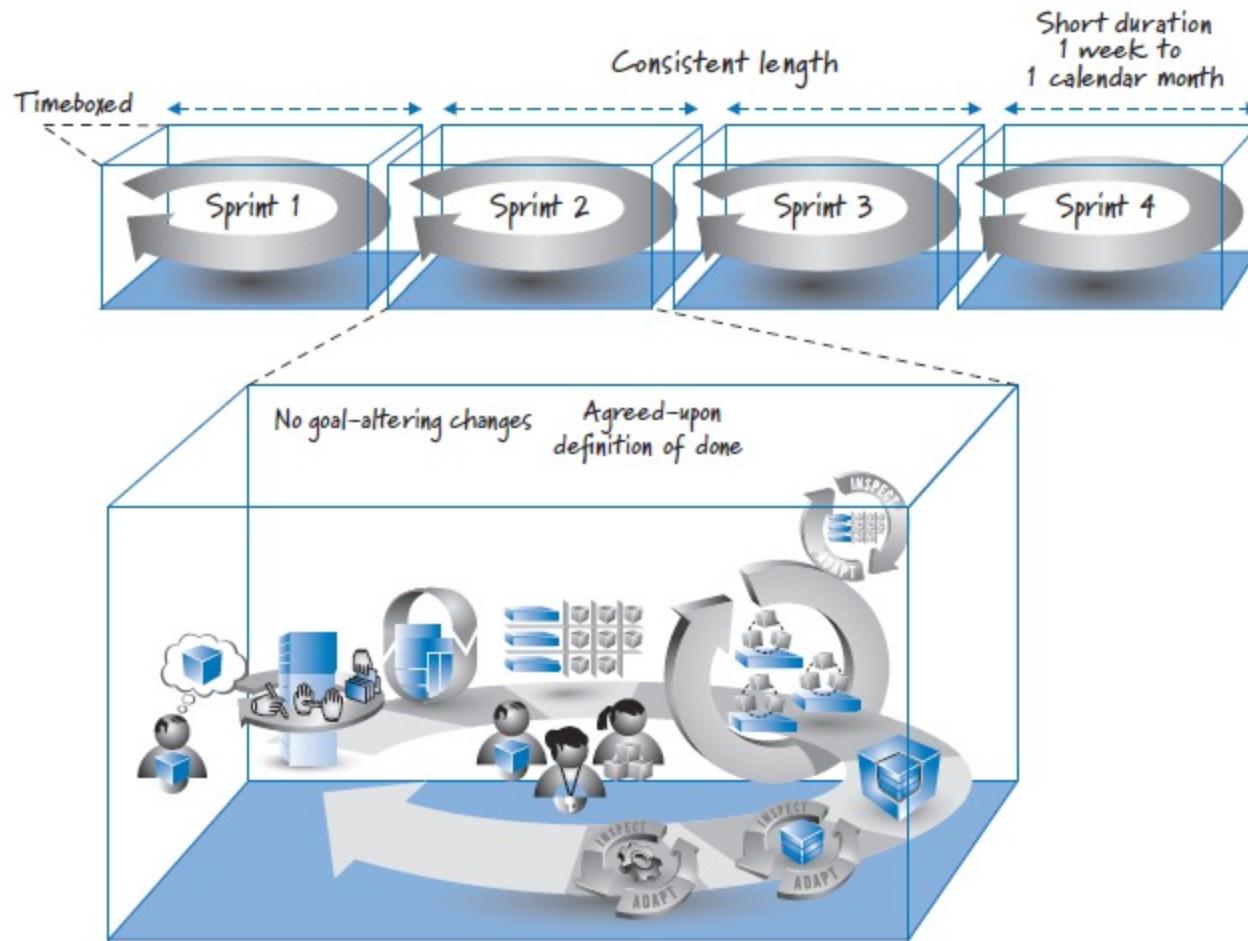


Scrum Elementi

Metodologije razvoja softvera

Sprint

- Predstavlja osnovu Scrum okruženja



Sprint - timebox

- Sprintovi moraju biti vremenski ograničeni, imaju tačan datum početka i kraja
 - Vremenski okvir se naziva **timebox**
- U okviru timebox-a od razvojnog tima se očekuje da održivim tempom završe dogovoreni skup zadataka koji je u skladu sa ciljem sprinta

Sprint - timebox

- Razlozi zbog kojih je bitno vremensko ograničavanje
 - Ograničavanje količine rada koja je u toku (WIP - work in process)
 - Timeboxing je tehnika za ograničavanje količine rada koja je u toku
 - WIP predstavlja inventar posla koji je započet, ali nije završen
 - Sam tim planira da radi na onim zadacima koje očekuje da će završiti
 - Loše upravljanje i organizacija sprinta može ozbiljno da utiče na projekat
 - Prioritizacija
 - Timeboxing prisiljava na prioritizaciji zadataka – ovo postiće rad na bitnim stvarima
 - Progres
 - Pomaže u demonstriranju progra time što se završavaju i validiraju bitni delovi do unapred poznatog datuma (kraj sprinta)
 - Smanjuje organizacione rizike pomjerajući fokus sa nepuzdanih izveštaja o učinku
 - Takođe pomaže razvojnog timu i klijentima da imaju bolji uvid u to šta je još preostalo do kraja projekta

Sprint - timebox

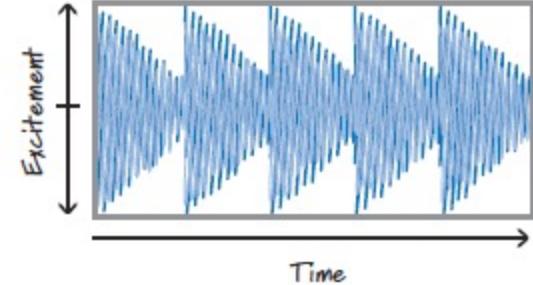
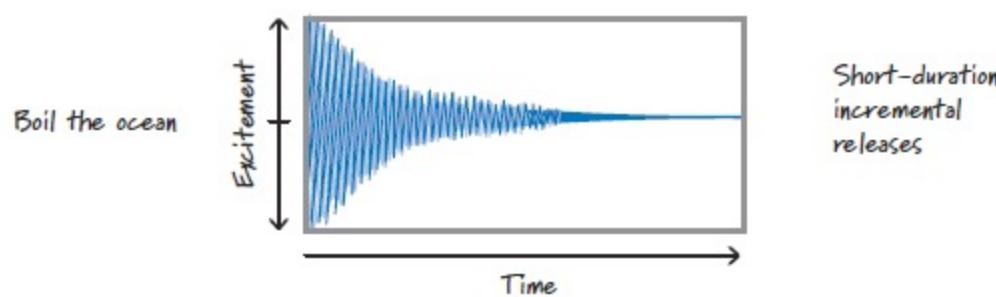
- Razlozi zbog kojih je bitno vremensko ograničavanje
 - Izbegavanje nepotrebnog perfekcionizma
 - Prisiljava razvojni tim da ne moraju baš svaku sitnicu uraditi do savršenstva
 - Motivisanje zatvaranja
 - Pokazalo se da se poslovi u većoj meri završavaju kada postoji jasan datum do kada se mora završiti
 - Poboljšavanje prediktivnosti
 - Iako nije moguće precizno predvideti količinu posla koju tim može uraditi u narednog godini, moguće je predvideti količinu posla u narednom sprintu koja može da se uradi

Sprint – kratko trajanje

- Razlozi zbog kojih je bitno kratko trajanje
 - Lakše planiranje
 - Lakše je planirati nekoliko nedelja unapred nego li nekoliko meseci
 - Sam postupak planiranja zahteva manje truda i mnogo je precizniji kod kraćeg trajanja
 - Brži feedback
 - Tokom svakog sprinta kreira se zadna verzija softvera koju klijent može da proba
 - U ranoj fazi razvoja moguće je uočiti neke nepoželjne solucije u razvoju softvera i zaobići ih

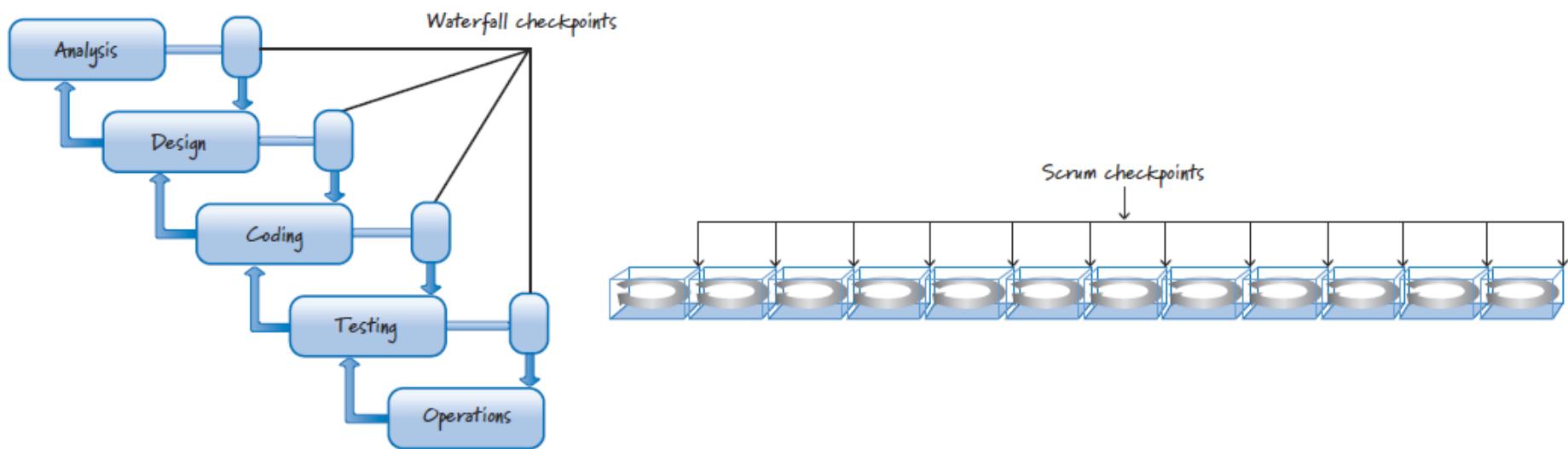
Sprint – kratko trajanje

- Razlozi zbog kojih je bitno kratko trajanje
 - Poboljšavaju povratak investicije
 - Omogućuju češće i brže isporuke funkcionalnog softvera – čime klijent ima priliku za ranije ostvarivanje prihoda od tog softvera
 - Kontrolisana/ograničena greška
 - Čak i ako sve što se uradi u sprintu bude pogrešno ne mora bitno da utiče na ishod čitavog projekta – *koliko lošeg može da se uradi u nedelji – dve?*
 - Entuzijazam
 - U prirodi čoveka da mu entuzijazam opada sa vremenom – na dužim projektima



Sprint – kratko trajanje

- Razlozi zbog kojih je bitno kratko trajanje
 - Češće kontrolne tačke
 - Mnogo više kontrolnih tačaka nego kod sekvencijalnog rada na projektu
 - U složenom okruženju lakše je funkcionalisati ako ima više kontrolnih tačaka u kojima je moguće izvršiti prilagođavanje



Sprint – konzistentno trajanje

- Trajanje svih sprintova bi trebalo da bude identično osim u specifičnim slučajevima
- Specijalni slučajevi
 - Pomeranje sa npr. četveronedeljnih na dvonedeljne sprintove radi češćeg feedback-a
 - Praznici, fiskalna godina, odmori
 - Očekivan period za novu verziju (release) – npt. nova verzija izlazi jednom nedeljno, a sprint je na dve nedelje
- Razlog za produženje sprinta ne može biti to što tim ne može da obavi čitav posao dogovoren za sprint
- Lakše se usklađuje ritam i olakšava planiranje

Sprint – nema promene cilja

- Jedno od važnijih pravila za sprint je da kada on započne nema promene cilja koji je definisan
- Šta je cilj sprinta?
 - Svaki sprint bi trebalo da se opiše ciljem sprinta koji predstavlja poslovni smisao i vrednost tog sprinta
 - Tokom faze planiranja tim treba da se dogovori oko toga šta će biti cilj i da se na osnovu toga odaberu iz product backlog-a koje će biti urađene u sprintu
- Obostrana posvećenost
 - Razvojni tima se posvećuje da će ispuniti cilj do kraja sprinta
 - Product owner se posvećuje da neće menjati cilj tokom sprinta
 - Balansira potrebnu da poslovanje bude adaptivno na izmene, a opet dozvoljava timu radi na nepromenljivom zahtevu

Sprint – nema promene cilja

- Pragmatičnost
 - Pravilo da nema promene cilja nije i zakon, tj. Scrum predviđa mogućnost da zbog novonastalih okolnosti ipak dođe do promene cilja
 - Npr. 1) konkurenčija lansira novi proizvod i u analiz se utvrdi da je potrebno promeniti cilj sprinta kako bi naš proizvod i dalje bio konkurentan, 2) u produpcionom sistemu se desila greška koju tim mora da ispravi

Sprint – nema promene cilja

- Abnormalan prestanak
 - Ako se desi situacija da je cilj sprinta postao potpuno besmislen, tim može odlučiti da sprint više nema smisla i predložiti product owner-u abnormalan prekid sprinta
 - Tim se sa product owner-om dogovara oko novog sprinta
 - U praksi ovo je dosta redak slučaj

Sprint - definicija završenosti

- Definicija završenosti – definition of done
- Kada se podrazumeva da je nešto završeno
- Obično se definiše lista zadataka (stavki) koje tim treba da uradi da bi se samtralo da je sprint završen
- Ova lista može da zavisi od:
 - Vrste projekta
 - Članova tima
 - Korišćenih tehnologija
 - Potencijalni prepreka koje mogu da utiču na projekat
- Lista može da evoluira tokom projekta
 - Dozvoljeno je menjati listu od sprinta do sprinta
 - Obično se ove izmene vrše u nekoliko početnih sprintova

Sprint - definicija završenosti

- Definicija završenosti odnosi se na deo proizvoda koji se razvija u okviru sprinta
- Kako se deo proizvoda u stvari sastoji od stavki iz product backlog-a, svaka od ovih stavki mora biti završena u skladu sa definicijom završenosti

Zahtevi - Uvod

Requirements and User Stories

- Requirements and User Stories
- U Scrum-u zahtevi su nešto o čemu se pregovara, nisu fiksirani
- Na zahteve se gleda kao nešto što ima visok stepen slobodo i što može da se menja da bi se što bolje postigao željeni poslovni cilj. Npr.
 - Ako ponestaje novca, neki malo manje bitni zahtevi se mogu odbaciti
 - Ako se tokom razvoja uvide neke nove činjenice koje utiču smanjenje prioritet zahteva, taj zahtev se može odbaciti
 - Ako se pojavi novi zahtev sa visokim prioritetom onda treba taj zahtev ubaciti u listu i posvetiti mu se

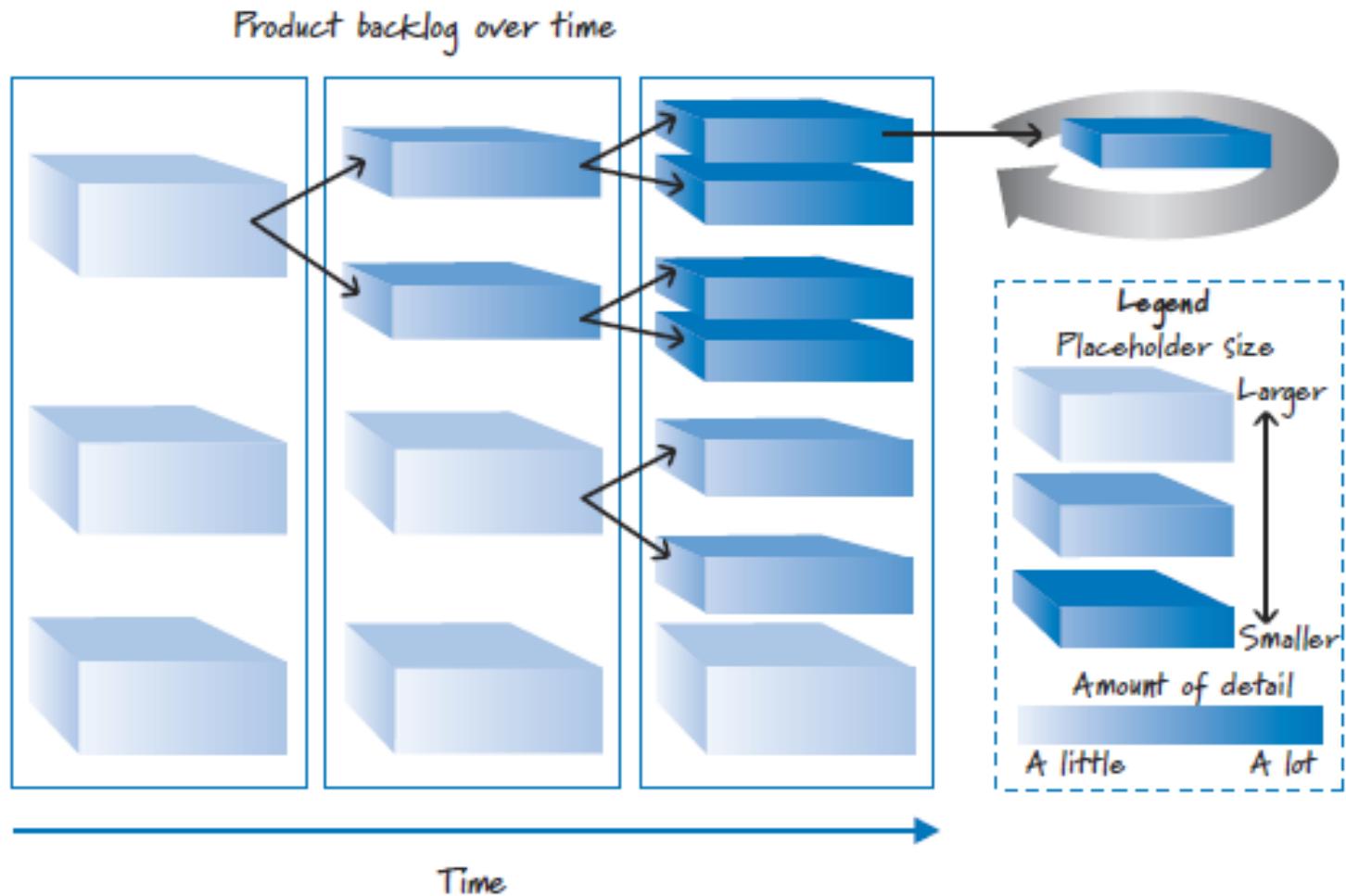
Zahtevi - Uvod

- Pokazalo se da kada se kreira neki novi proizvod koji ima visok stepen inovacija učestalo je da promena zahteva je jako visoka
 - Nije moguće na samom početku kreirati sve zahteve koliko god se trudili
- Scrum ne zahteva da se svi zahtevi deljano razrade na samom početku projekta
 - Ovim se štedi na vremenu i novcu
 - Očekuje se da će tokom vremena biti izmena, odnosno da će se neki zahtevi pojednostaviti tokom razvoja

Zahtevi - Uvod

- Umesto da se u Product backlog-u (PB) kreira gomila detaljnih stavki (items) na samom početku kreira se manji skup stavki koji će vremenom evoluirati
- Svaka stavaka PB-a vezana je za neku poslovne ciljeve (business value) koja se želi implementirati

Zahtevi - Uvod



Zahtevi - Uvod

- Inicijalno stavke PB-a su velike
 - Predstavljaju širok opseg poslovnih ciljeva sa jako malo detalja
- Tokom vremena ove stavke će tokom konverzacije između učesnika biti redefinisane u kolekciju manjih i detaljnijih stavki
- Na kraju će PB stavke biti dovoljno male i dovoljno detaljne da mogu da se realizuju u okviru sprinta
 - Čak i tokom sprinta će se neki detalji razraditi u komunikaciju razvojnog tima i Product owner-a

Zahtevi - Uvod

- Scrum ne definiše standardni format za predstavljanje stavki PB-a
- U praksi mnogi timovi koriste „korisničke priče“ (*user stories*) za reprezentaciju
- Osim *user stories* koriste se i slučajevi korišćenja ili neke druge prilagođene tehnike

Zahtevi - Konverzacija

- Jedan od bitnih problema kod sekvensijalnog razvoja nastao je zbog prevelikog oslanjanja isključivo na pisane zahteve
- U Scrum-u konverzacija se koristi kao ključna poluga za razumevanje i pojašnjenje navedenih zahteva
- Konverzacija i dalje ne zamenuje u potpunosti dokumentaciju
 - Oni koji žele ili moraju da kreiraju specifikaciju zahteva i dalje to mogu uraditi na osnovu stavki PB i detaljnog opisa vezanog za njih

Zahtevi – Progresivno usavršavanje

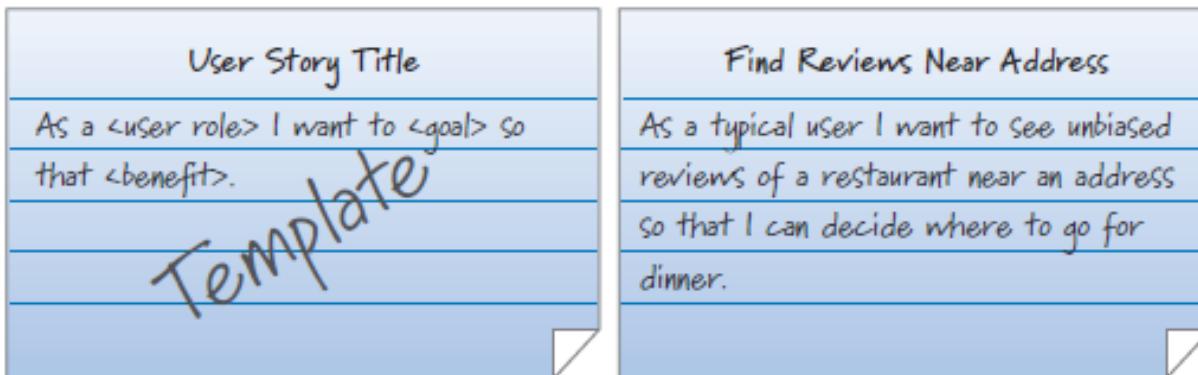
- Kod sekvencijalnog razvoja svi zahtevi moraju biti na istom nivou detalja u isto vreme, tj. ne bi smelo da se naknadno dodaju neki detalji za zahtev
- Ovo izaziva nekoliko problema:
 - Moraju se predvideti svi detalji u ranoj fazi razvoj kada se, po pravilu, najmanje zna o proizvodu
 - Svi zahtevi se tretiraju jednako bez obzira na prioritet – što znači da će se možda potrošiti vreme na nešto što se možda neće implementirati
 - Kreira se velika baza zahteva koja će se verovatno vrlo često tokom vremena menjati što je dosta skup proces
 - Smanjuje se verovatnoća, da koristeći konverzaciju, se pojasne zahtevi pošto su oni već definisani

Zahtevi – User stories

- Format za predstavljanje poslovni ciljeva u Product backlog-u
- Nastaju na način koji ih čini razumljivim i poslovnim i tehničkim ljudima
- Strukturno su obično jednostavne
- Mogu biti napisane na različitim nivoima granularnosti i mogu progresivno da se modifikuju
- Jeden način za opis/definisanje user stories – 3C:
 - Cards (kartice)
 - Conversation (konverzacija)
 - Confirmation (potvrda)

Zahtevi – User stories

- Ideja **kartica** je da korisnici na relativno malom parčetu papira napišu jednu user story
- Najčešći template za pisanje je da se navede **uloga korisnika** koji želi da postigne određeni **cilj** i šta je **benefit** postizanja tog cilja
- Namena kartice nije da se opišu svi detalji user story već da iskaže suštinu
- Predstavlja osnovu za neku dalju diskusiju



Zahtevi – User stories

- Detalji zahteva se razotkrivaju i razrađuju u **konverzaciji** između razvojnog tima, product owner-a i klijenata
- Konverzacija po pravilu nije jednokratni događaj već nešto što će da traje
 - Jedna konverzacija tokom definisanja user story
 - Druga konverzacija tokom razrade user story
 - Treća tokom procene
 - Četvrta tokom planiranja sprinta
 - Peta tokom implementiranja i testiranja
 - ...
- Iako je konverzacija verbalna može se dopuniti odgovarajućom dokumentacijom

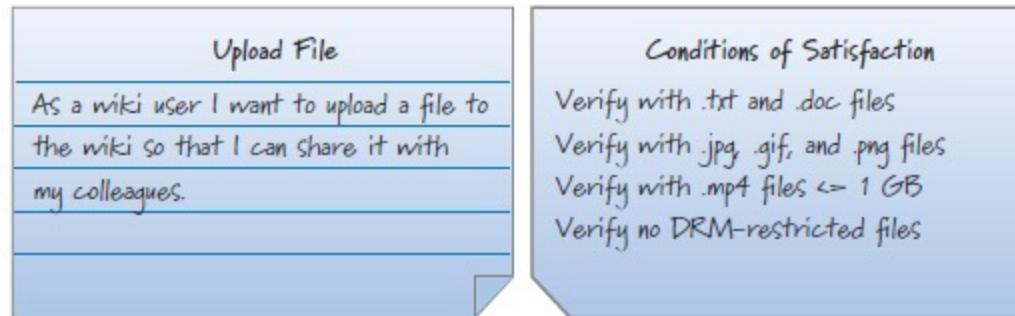
Johnson Visualization of MRI Data
As a radiologist I want to visualize MRI data using Dr. Johnson's new algorithm. For more details see the January 2007 issue of the Journal of Mathematics, pages 110-118.

Zahtevi – User stories

- User story sadrži **potvrdu** u form uslova za ispunjenost (zadovoljenje)
- Ovi uslovi predstavljaju kriterijume prihvatljivosti koji pojašnjavaju željeno ponašanje
- Koriste se od strane razvojnog tima radi boljeg razumevanja šta treba da se realizuje i testira i od strane product owner-a kako bi potvrdio da je ta user story implementirana u skladu sa njegovim zahtevima

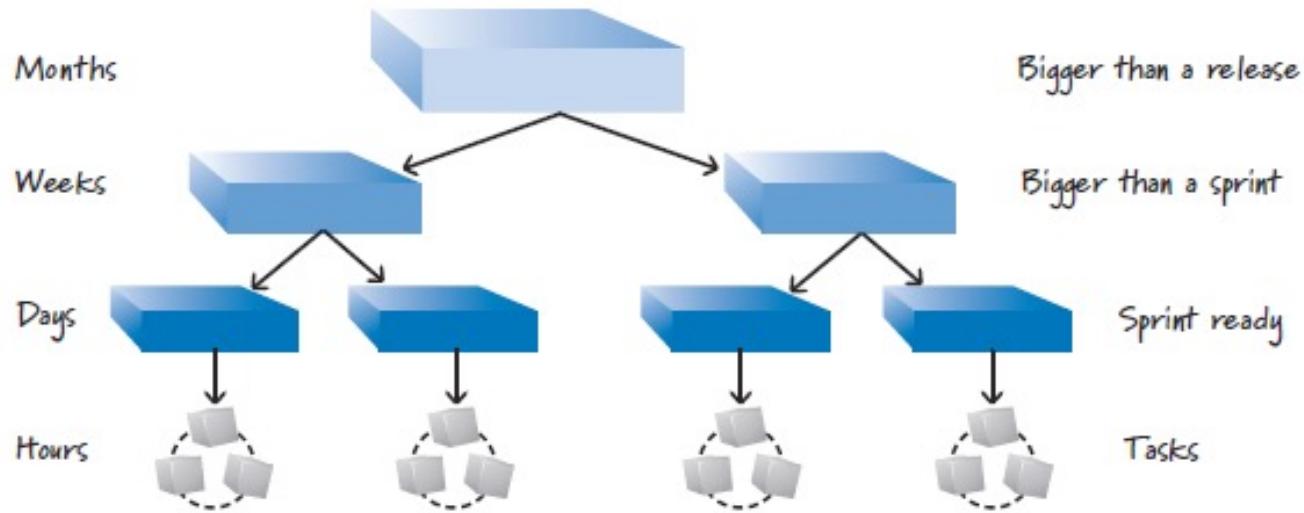
Zahtevi – User stories

- Uslovi se obično navode na poledžini kartice
- Mogu se izraziti kao testovi prihvatljivosti (acceptance tests) visokog nivoa
 - Nisu jedini testovi za proveru funkcionalnosti iz user story
 - Sa strane product owner-a ovi testovi su važni jer predstavljaju dokaz da je nešto implementirano kako treba
 - Zgodni za definisanje inicijalnih user story-ja i za njihovo redefinisanje
 - Pristup baziran na testovima se zove još i **specification by example** ili **acceptance-test-driven development (ATTD)**



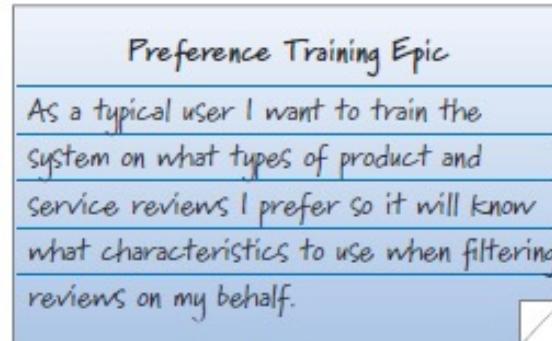
Zahtevi – User stories

- Korišćenje user stories iste veličine tokom čitavog procesa razvoja nekog softvera nije pogodno
 - Npr. krake user stories su pogodne za sprintove, ali ne i za ne planiranja na visokom nivou
- Po pravili user stories se pišu na različitim nivoima apstrakcije
 - Obično što je nivo apstrakcije viši to je potrebno više vremena za implementaciju - često više meseci i često opisuju jedan čitav release



Zahtevi – User stories

- User stories na najvišem nivu apstrakcije se zovu još i (epovi) **epics**
 - Opisuju globalnu sliku ono što se želi postići
 - Epic nikad ne bi trebao da bude polazna osnova za sprint, već ga treba posmatrati kao nešto što predstavlja osnovu za kolekciju mnogo manjih i detaljnijih user stories



Zahtevi – User stories

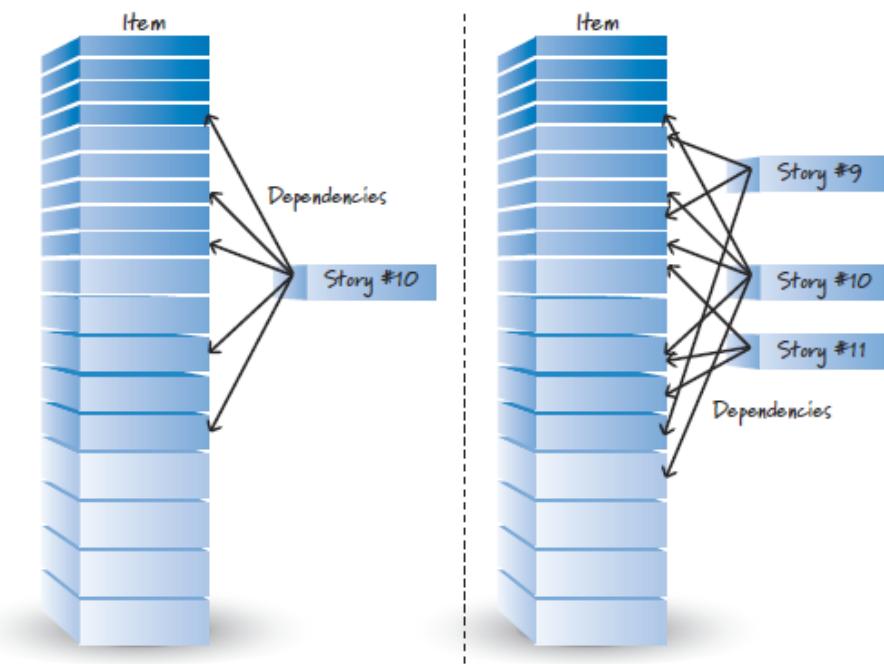
- User stories u trajanju od nekoliko nedelja zovu se još i **features**
- Najkraća forma se zove još i **stories/sprintable stories/implementable stories**
- Pojedini timovi koriste i pojam **teme (theme)** koji se predstavlja kolekcija povezanih user stories
 - Zgodan način da se vidi šta pripada istoj funkcionalnoj oblasti
- **Zadaci (tasks)** su nivo ispod user stories
 - Obično predstavljaju nešto na čemu će raditi jedan ili dva člana tima
 - Njihova realizacija se meri u satima
 - Nisu isto što i user stories, tako da ne bi trebalo da se definišu kada i user stories
 - Task obično opisuje kako nešto napraviti, a ne šta napraviti

Zahtevi – User stories

- Kako napisati dobre user stories?
- Jedna preporuka je INVENT kriterijum koji se prilikom evaluacije koliko user stories odgovaraju svojoj nameni:
 - Independent (nezavisnost)
 - Negotiable (mogućnost pregovaranja/dogovaranja)
 - Valuable (korisnost)
 - Estimatable (procenljivost)
 - Small (odgovarajuće dimenzionisan)
 - Testable (mogućnost testiranja)

Zahtevi – User stories

- **Independent (nezavisnost)**
 - Koliko god je moguće user stories bi trebalo da su međusobno nezavisne ili bar što labavije povezane
 - User stories koje imaju visok stepen zavisnosti komplikuju procenu, definisanje prioriteta i planiranje



Zahtevi – User stories

- **Negotiable (mogućnost dogovaranja/pregovaranja)**
 - User stories nisu pisani ugovor u formi unapred definisanih korisničkih zahteva već predstavljaju osnovu za nešto o čemu će se dalje pregovarati i diskutovati
 - Dobre user stories jasno definišu koja poslovna funkcionalnost se želi realizovati i zašto, sa druge strane ostavljaju dovoljno prostora svim učesnicima da se dogovore oko detalja
 - Najčešći slučaj gde se dogovaranje narušava je kada product owner navede kako treba neka user story da se implementira

Zahtevi – User stories

- **Valuable (korisnost)**
 - Svaka user story bi trebala da ima neku vrednost ili klijentu ili krajnjim korisnicima softvera ili oboma
 - Ako od neke user story korist nema ni klijent ni krajnji korisnik onda ona ne bi trebalo da se nađe u product backlog-u
 - Postavlja se pitanje šta sa user stories koje imaju neku vrednost razvojnom timu, ali ne i klijentu i korisnicima – **tehničke user stories**
 - Ključni problem kod ovakvih user stories je što treba ubediti product owner-a u njihov svrshishodnost, tj. objasniti mu zašto treba da finansira i to
 - U praksi tehničke user stories se obično ne smeštaju u product backlog, već se obično predstavljaju kao task-ovi



Zahtevi – User stories

- **Estimatable (procenljivost)**
 - Svaka user story bi trebala da bude procenljiva od strane razvojnog tima koji će izvršiti dizajn, implementaciju i testiranje
 - Procena obezbeđuje indikator o veličin user story, a time i indikatore o vremenu i trošku potrebnom za realizaciju
 - Veličina user story je jako biran informacija za Scrum tim
 - Ako tim nije u stanju da proceni veličinu user story onda je ona ili jako velika, ili dvosmislena (neprecizna) ili tim nema dovoljno znanja da bi izvršio adekvatnu procenu
 - Ako je suviše velika tim treba zajedno sa produc owner-om da je dekomponuje na više manjih
 - Ako je neprecizna treba da se rasprave elementi koju je čine nepreciznom/dvosmislenom
 - Ako tim nema dovoljno znanja onda je potrebna dodatna aktivnost po pitanju pojašnjenja tog user story-ja

Zahtevi – User stories

- **Small (odgovarajuće dimenzionisan)**
 - Veličina user stories zavisi od nivoa apstrakcije
 - User stories namenjeni za sprintove treba da budu dimenzionisani u skladu sa dogovorenim trajanjem sprintova
 - Poželjno je da user story traje kraće od samog sprinta
 - Uvek je bolje imati u jednom sprintu više user stories koje traju kraće, nego li imati jednu user story koja traje jednakо kao i sam sprint pošto je tada rizik nezavršetka user story veći

Zahtevi – User stories

- **Testable (mogućnost testiranja)**
 - User stories bi trebalo da mogu da se tesitaju u Bulovom smislu: testovi su ili prošli ili nisu
 - User story koji može da se testira ima dobar kriterijum prihvatanja (**acceptance criteria**)
 - Bez mogućnosti testiranja ne može se znati da li je određena user story realizovana ili ne na kraju sprinta
 - Za neke user stories ne postoji praktičan način za testiranje
 - Npr. Od sistema se zahteva 99.999% uptime u produkciji

Zahtevi – User stories

- **Nefunkcionalni zahtevi predstavljaju ograničenja na nivu čitavog sistema**
- Mogu, a ne moraju da se predstavljaju kao user stories
- Utiču na dizajn i testiranje većine user stories u product backlog-u
- Svaki funkcionalni zahtev je nešto što se obično uključuje u definiciju završenosti

Internationalization

As a user I want an interface in English, a Romance language, and a complex language so that there is high statistical likelihood that it will work in all 70 required languages.

Web Browser Support

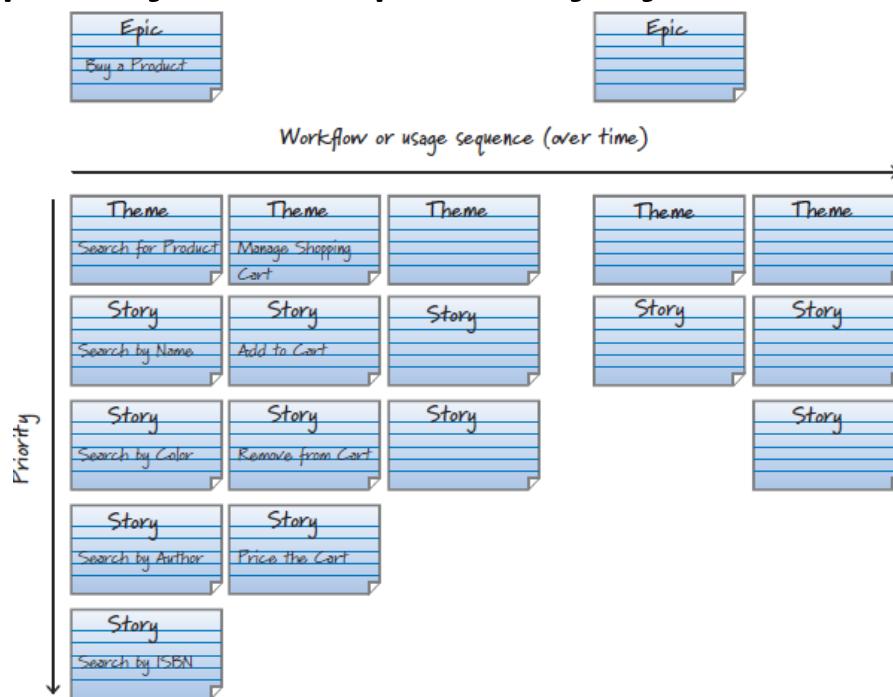
System must support IEB, IE9, Firefox 6, Firefox 7, Safari 5, and Chrome 15.

Zahtevi – User stories

- Kako se formiraju user stories?
- U tradicionalnom pristupu zahtevi se formiraju tako što se korisnici pitaju šta žele – u praksi se ovo pokazalo kao loš pristup
- Drugi pristup je uključiti korisnike u tim koji odlučuje šta treba da se napravi i koji konstantno razmatra šta se pravi
 - Obično se sprovode **user-story-writing workshop-ovi**
 - Na ovim workshop-ovima učestvuje i product owner, ScrumMaster, razvojni tim, klijenti i neka druga strana (ako treba)
 - Traju od nekoliko sati do nekoliko dana
 - Od workshop-a se ne očekuje da se odjednom ne generiše kompletna lista user stories, već obično ima specifični fokus
 - Npr. Na workshop se razmatra šta treba da ide u sledeći release
 - Na prvom workshop-u se obično i određuju uloge u organizaciji sa ciljem da se definišu uloge koje se u delu uloge u user stories (“As a <user role>, I want to . . .”)
 - Tokom samog workshop-a nema standardnog načina za generisanje user stories
 - Koristi se i top-down i bottom-up pristup

Zahtevi – User stories

- **Story mapping** je tehnika koja se bazira na korisnički orijentisanoj perspektivi za kreiranje user stories
- Osnovna ideja je da se korisničke aktivnosti visokog nivoa dekomponuju u workflow koji se potom dekomponuje u skup detaljnijih zadataka



Zahtevi – User stories

- Osnovna razlika između **user-story-writing workshop-a** i **story mapping-a** je što je tokom workshop-a fokus primarno na genrisanju user stories, a ne na definisanju njihovog prioriteta
- Tehnika story mapping-a se može iskoristiti kao komplement workshop-u da pomogne u definisanju prioriteta i vizuelizaciji prioriteta user stories

Product Backlog

- Lista željenih funkcionalnosti sa prioritetima
- Obezbeđuje podršku za centralizovano i prošireno razumevanje šta se razvija i redosled u kome treba da se razvija
- Dostupan je svim učesnicima projekta

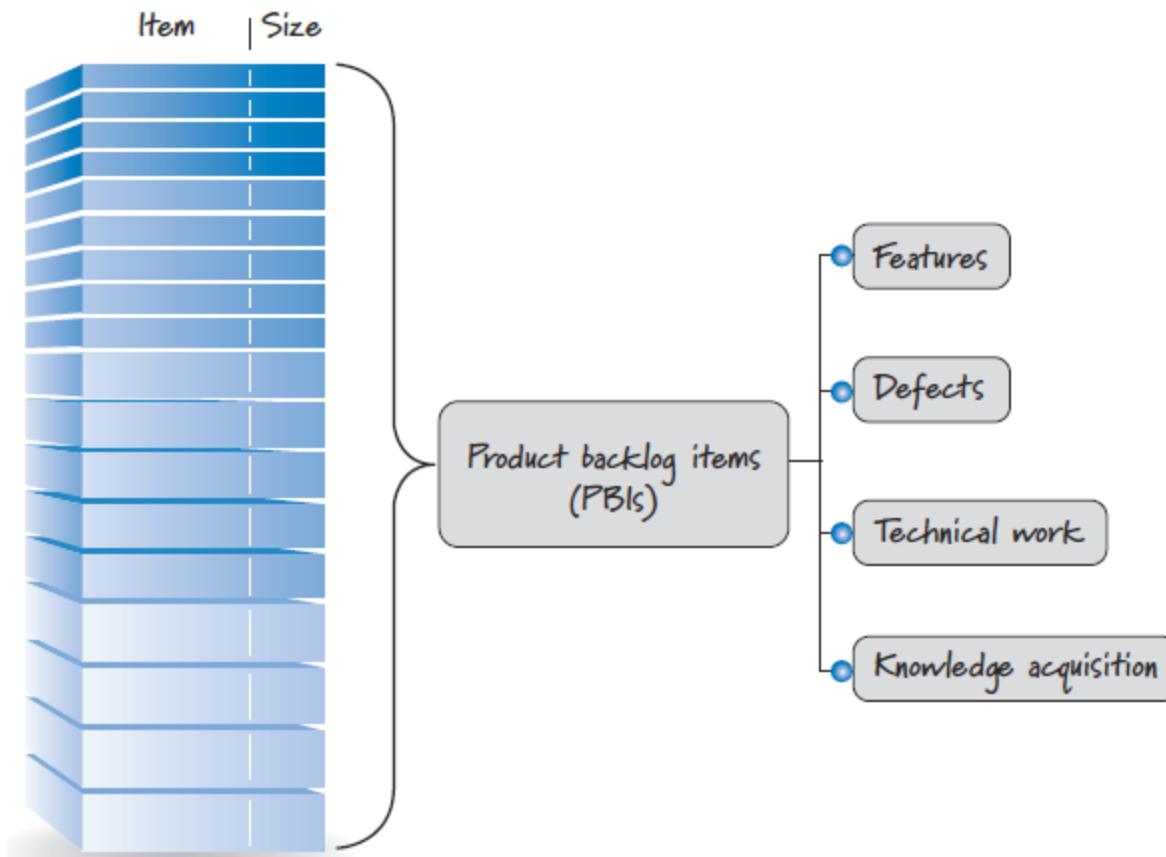


Product Backlog - Stavke

- Product Backlog sastoji se od stavki (Product Backlog Items - PBIs)
- Većina stavki su funkcionalnosti (**features**) koje imaju neku vrednost za klijenta
 - Potupno nove funkcionalnosti ili postojeća koja treba da se promeni (**change**)
- Stavke su najčešće su napisane kao user stories
- Osim funkcionalnosti PBIs mogu da opisuju i neke nedostatke koje treba popraviti (**defects**), tehnička poboljšanja (**technical work**), akviziciju znanja (**knowledge acquisition**)

Product Backlog - Stavke

- Vrste stavki Product Backlog-a



Product Backlog - Stavke

- Primeri stavki

PBI Type	Example
Feature	As a customer service representative I want to create a ticket for a customer support issue so that I can record and manage a customer's request for support.
Change	As a customer service representative I want the default ordering of search results to be by last name instead of ticket number so that it's easier to find a support ticket.
Defect	Fix defect #256 in the defect-tracking system so that special characters in search terms won't make customer searches crash.
Technical improvement	Move to the latest version of the Oracle DBMS.
Knowledge acquisition	Create a prototype or proof of concept of two architectures and run three tests to determine which would be a better approach for our product.

Product Backlog – Kvalitet

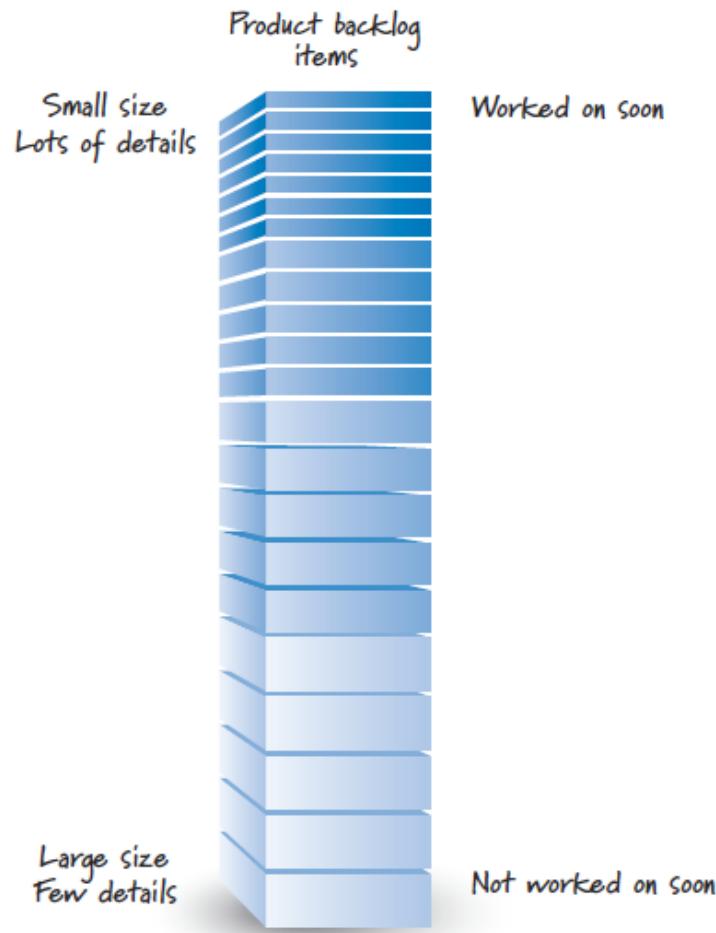
- Jedna preporuka za formiranje kvalitetnog PB-a je DEEP kriterijum
 - Detailed appropriately (adekvatan nivo detalja)
 - Emergent (promenljiv)
 - Estimated (procenjen)
 - Prioritized (dodeljen prioritet)
- Slično kao što je INVENT kriterijum pogodan za ocenjivanje user stories, DEEP kriterijum je zgodan za proveru da li je PB strukturiran na dobar način

Product Backlog – Kvalitet

- Detailed appropriately (adekvatan nivo detalja)
 - Ne očekuje se da sve stavke PB u istom trenutku imaju isti nivo detalja
 - PBIs na kojima se planira uskoro raditi trebalo bi da se nalaze negde na vrhu PB-a, da budu relativno male (da se mogu realizovati u okviru jednog sprint-a) i da budu veoma detaljne
 - PBIs na kojim se uskoro neće raditi nalaze se pri dnu PB i po pravilu su relativno velike i malo detalja
 - Kako se bliži rad na nekoj većoj stavci (npr epu) ona se dekomponuje na više manjih koje se potom detaljno razrađuju
 - Ovo treba da se vrši po *just-in-time* principu, tj. dekompozicija i detaljan opis se vrši baš kad je potrebno, ne mnogo ranije ili kasnije

Product Backlog – Kvalitet

- Detailed appropriately (adekvatan nivo detalja)



Product Backlog – Kvalitet

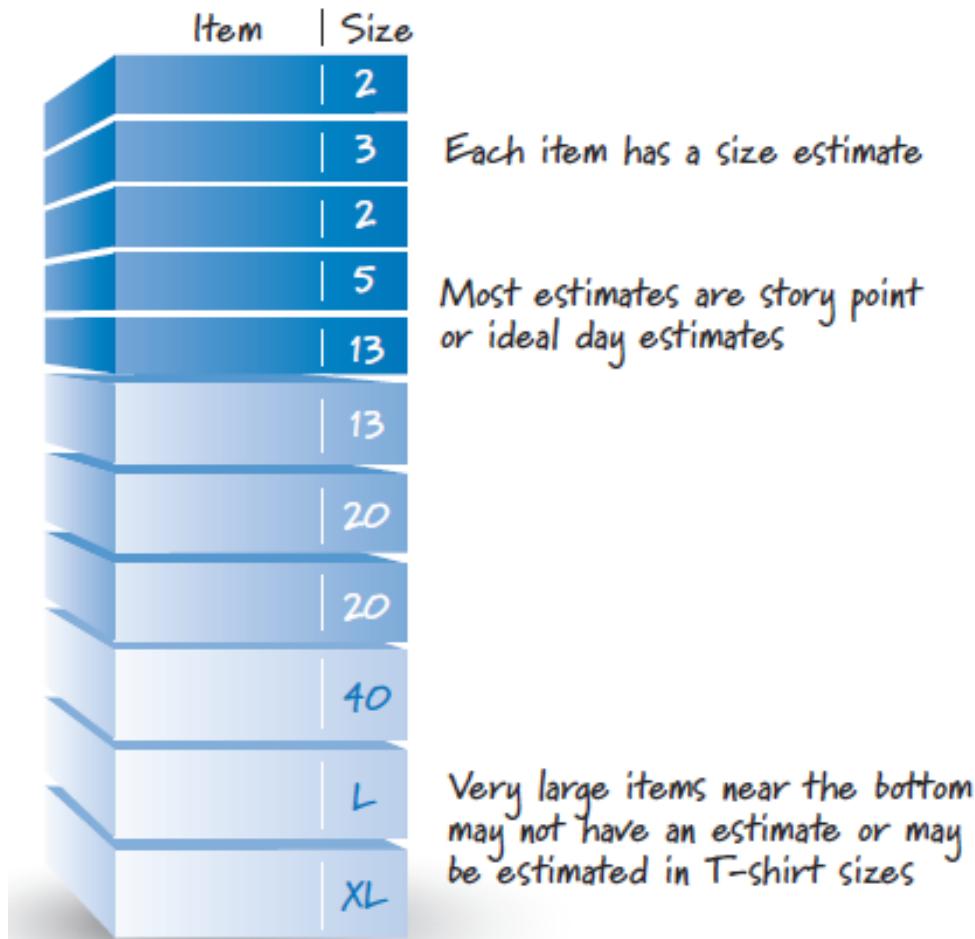
- Emergent (promenljivost)
 - Sve dok se proizvod razvija ili održava PB nikad nije kompletan ili zatvoren već se konstantno menja na osnovu različitih faktora: ekonomskih, tehničkih ...
 - Struktura PB se konstantno menja tokom vremena
 - Dodaju se nove stavke
 - Redefinišu se postojeće
 - Product Owner vrši rebalan i promenu prioriteta
 - ...

Product Backlog – Kvalitet

- Estimated (procenjen)
 - Za svaku stavku PB-a procenjuje se odgovarajuća vrednost (veličina) koj odgovara trudu potrebno za realizaciju te stavke
 - Product owner koristi ove procene kao jednu od stvari prilikom određivanja prioriteta PBIs, a time i njihovog položaja u PB-u
 - Većina stavki PB-a se procenjuju u **poenima ili danima** (*o ovome više priče kasnije ...*)
 - Procne bi trebalo da budu razumno tačne, ne moraju biti preterano precizne
 - Stavke na vrhu PB je moguće mnogo preciznije proceniti nego li stavke na dnu prvenstveno jer su manje i imaju više detalja
 - Veće stavke se često ne procenjuju ili se koriste grublje ocene (M, L, XL, XXL, ...)

Product Backlog – Kvalitet

- Estimated (procenjen)

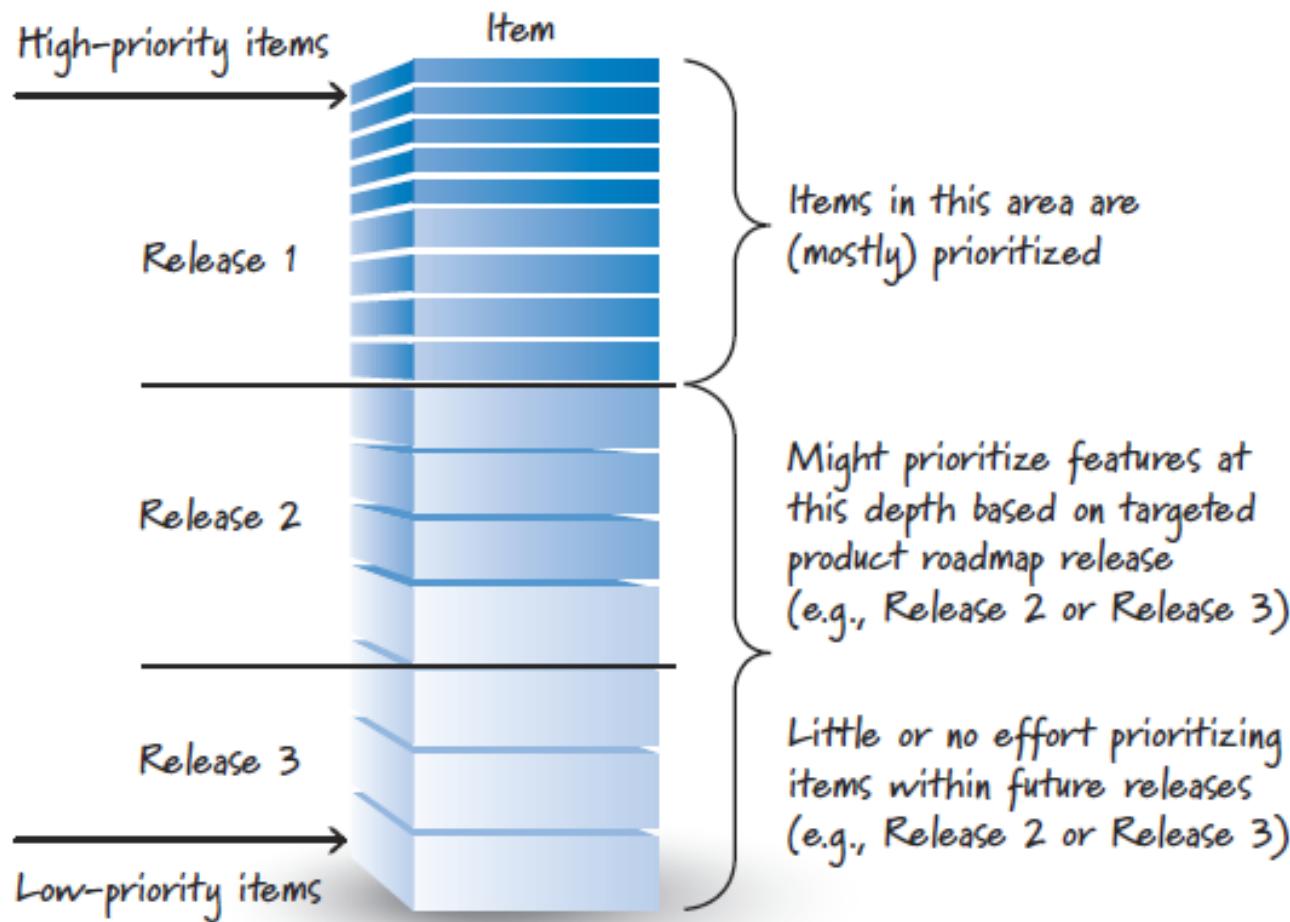


Product Backlog – Kvalitet

- Prioritized (dodeljen prioritet)
 - Iako se PB obično definiše kao lista stavki sa dodeljenim prioritetima, nije realno da će svim stavkama u PB-u biti dodeljen prioritet
 - Obično se prioritet dodeljuje stavkama za koje s očekuje da će biti realizovane u nekoliko narednih sprintova
 - U praksi se pokazalo zgodno da se prioritet dodeli stavkama koje se odnose na tekući release, a za sve naredne realeas-ove prioritet može u grubo da se definiše ili se ne definiše uopšte
 - Može se desiti da naredno release-a neće ni biti pa je onda uzaludno potrošeno vreme i novac

Product Backlog – Kvalitet

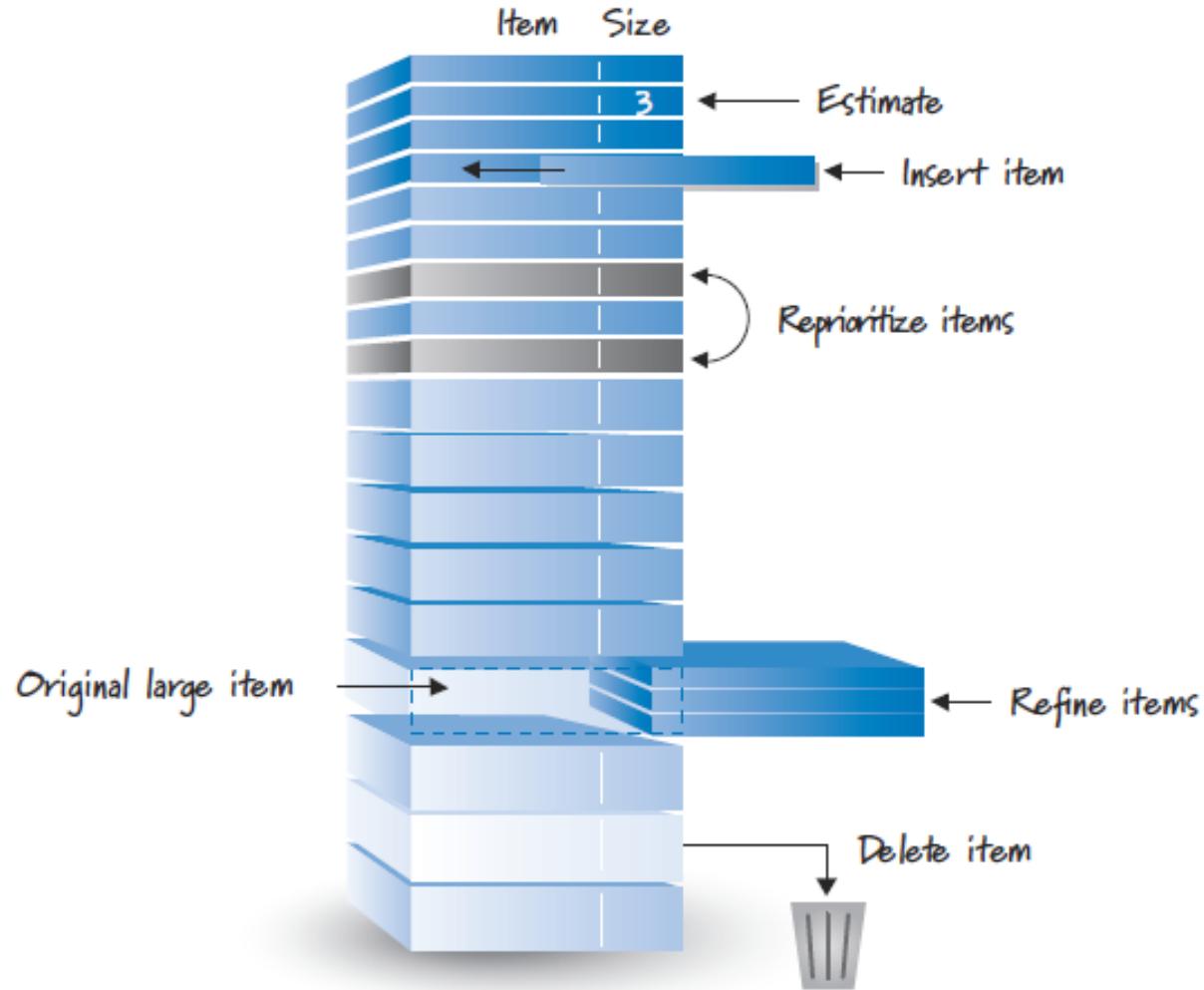
- Prioritized (dodeljen prioritet)



Product Backlog - Grooming

- Da bi se postigao DEEP princip PB se mora redovno administrirati, upravljati, organizovati što se u literaturi obično zove product backlog **grooming**
- Grooming se odnosi na tri osnovne aktivnosti kreiranje i ažuriranje (dodavanje detalja) PBI, procena PBI i dodeljivanje prioriteta PBI

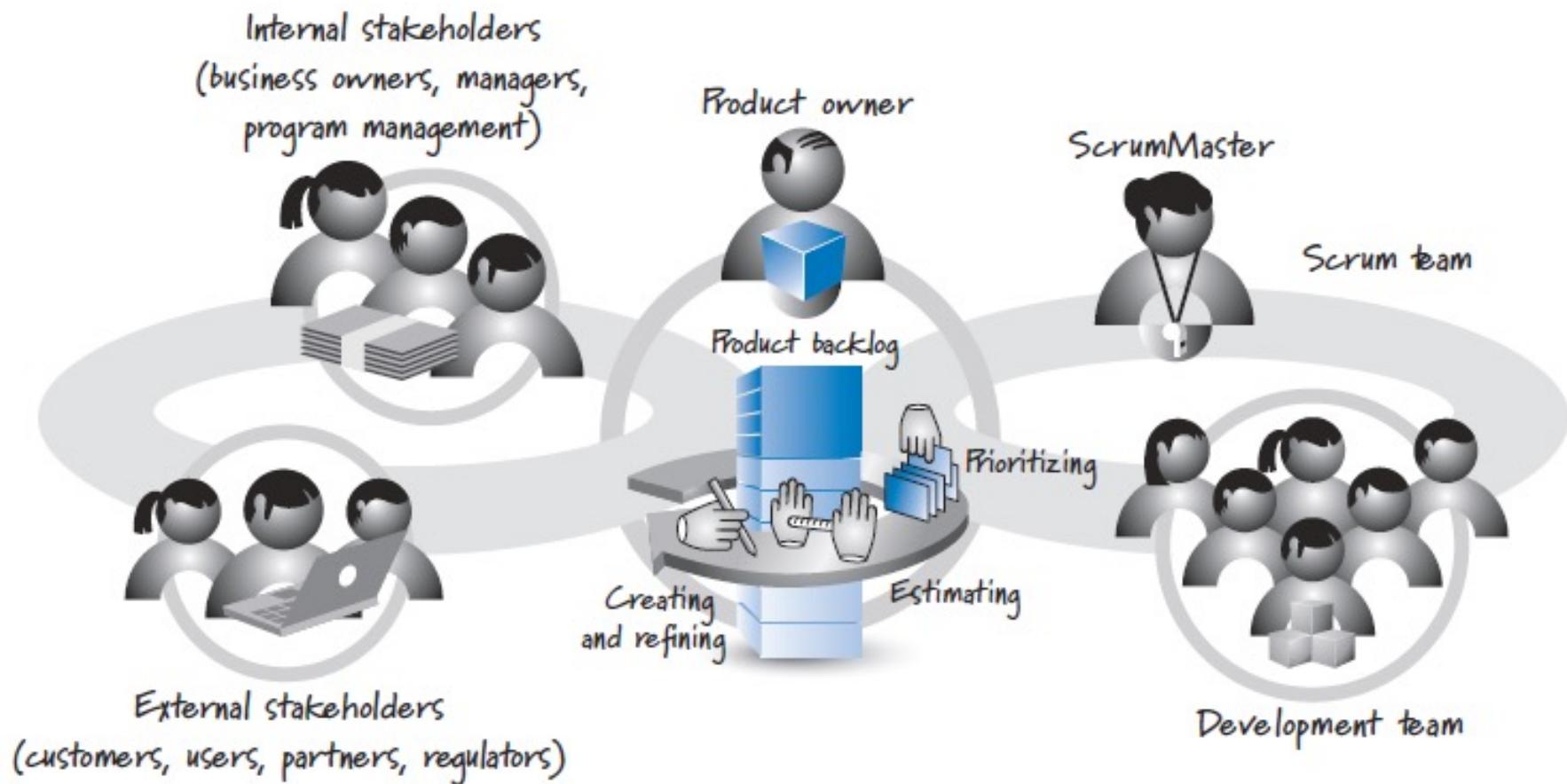
Product Backlog - Grooming



Product Backlog - Grooming

- Ko sprovodi grooming?
 - Pored Product Ownera učestvuju i klijenti, razvojni tim, ScrumMaster i eventualno i neki drugi učesnici koji imaju interes u projektu ili koji su važni za projekat
 - Učešćem svih ovih uloga stiče se bolje razumevanje o projektu i smanjuje se verovatnoća da je nešto pogrešno protumačeno
- Obično je Product Owner osoba koja vodi čitav postupak i neko ko daje finalnu odluku za svaku aktivnost u grooming procesu
 - Ipak ove odluke bi trebalo da se donešu konsenzusom svih učesnika i da na kraju postoji razumevanje svih učesnika zašto su takve odluke donešene
- Što se tiče klijenata oni za ovaj postupak treba da alociraju dovoljn vremena da može kvalitetno da se obavi
 - Vreme zavisi od same organizacije i tipa projekta
- Razvojni tim obično odvaja do 10% vremena svakog sprinta na grooming aktivnost

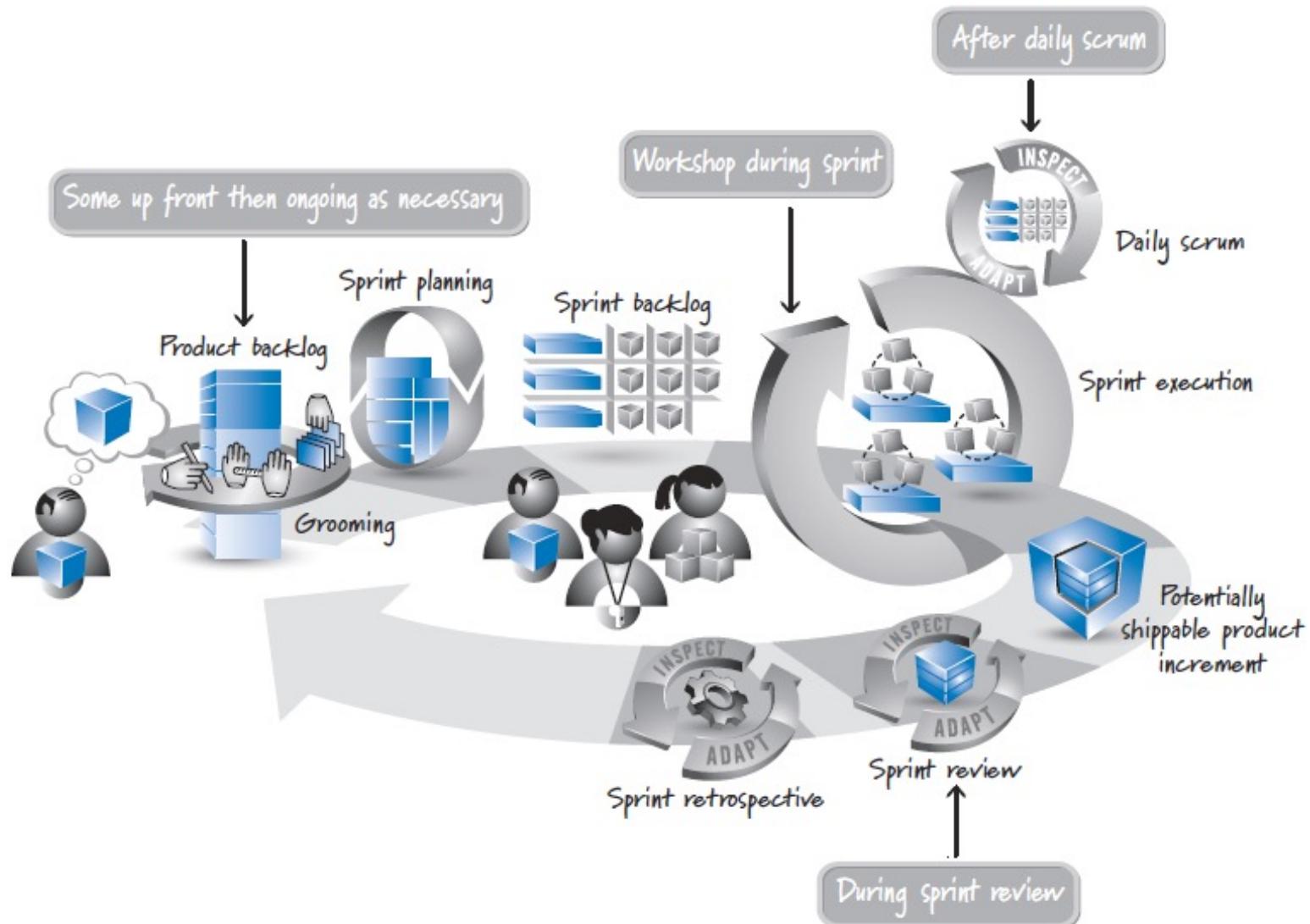
Product Backlog - Grooming



Product Backlog - Grooming

- Kad treba da se vrši grooming?
 - Scrum kaže da grooming treba da se sprovede, ali ne specificira kada
 - Pošto se u Scrum-u očekuje da se PB može konstantno menjati potrebno je obezbediti da grooming aktivnosti budu jedan od ključnih delova kako upravljati zadatim poslom, tj. omogućiti da se grooming aktivnosti obave u okviru svake faze Scrum-a
 - Neki timovi grooming obavljaju tokom sprinta, umesto da to obave u jednom vremenskom bloku – inkrementalni grooming
 - U ovom slučaju ne moraju svi biti uključeni u grooming

Product Backlog - Grooming



Product Backlog–Definicija spremnosti

Definition of Ready

- Grooming će obezrediti da stavke PB koje su na vrhu budu spremne za sprint
- Neki Scrum timovi su formalizovali ovu ideju tako što su uveli **definiciju spremnosti**
- Po definicijom spremnost podrazumeva se lista zahteva koje stavka PB treba da ispuni da bi bila sprmena za sprint
- Pokazao se da ova lista pozitivno utiče na šansu da će razvojni tim ispuniti cilj sprinta

Product Backlog–Definicija spremnosti

- Primer liste za definiciju spremnosti

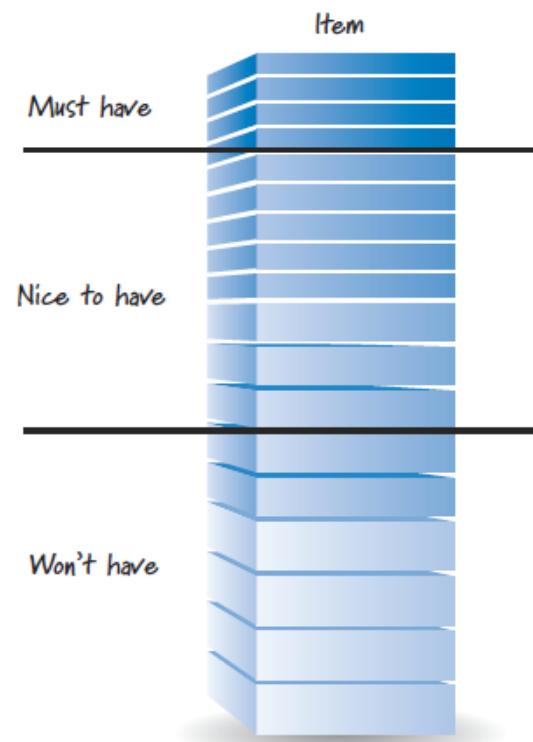
Definition of Ready	
<input type="checkbox"/>	Business value is clearly articulated.
<input type="checkbox"/>	Details are sufficiently understood by the development team so it can make an informed decision as to whether it can complete the PBI.
<input type="checkbox"/>	Dependencies are identified and no external dependencies would block the PBI from being completed.
<input type="checkbox"/>	Team is staffed appropriately to complete the PBI.
<input type="checkbox"/>	The PBI is estimated and small enough to comfortably be completed in one sprint.
<input type="checkbox"/>	Acceptance criteria are clear and testable.
<input type="checkbox"/>	Performance criteria, if any, are defined and testable.
<input type="checkbox"/>	Scrum team understands how to demonstrate the PBI at the sprint review.

Product Backlog – Upravljanje tokom Flow Management

- Neodređenost/neizvestnost u razvoju softvera teško može da se izbegne
- U velikoj meri product backlog smanjuje tu neodređenost, ali retko kad može da je svede na nulu
- Uloga product backlog sa stanovišta upravljanje neodređenosti posebno je važna za podršku:
 - Release Flow Management
 - Sprint Flow Management

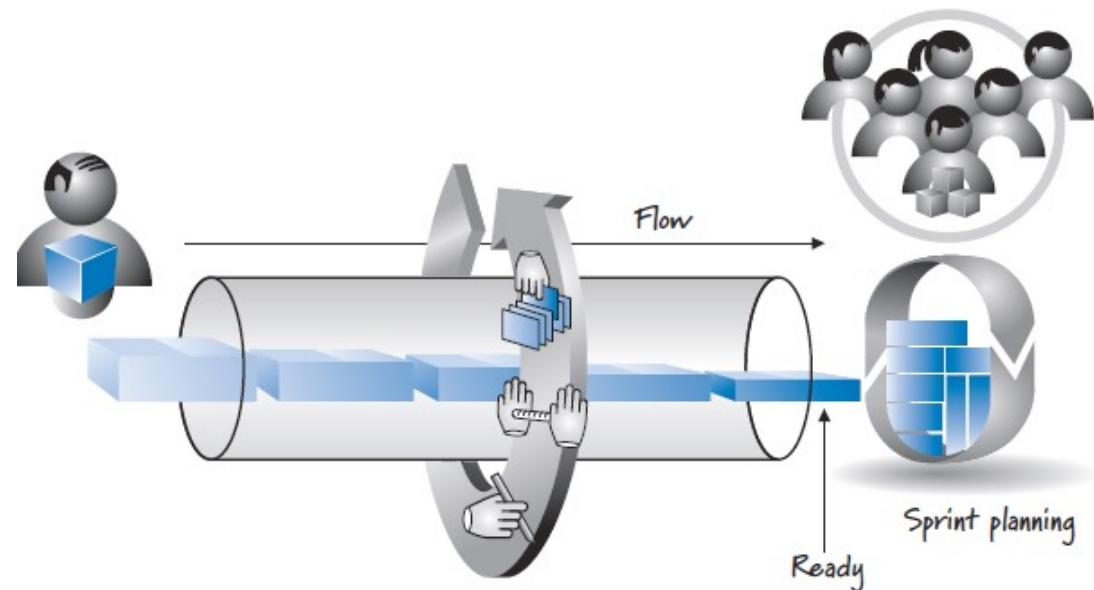
Product Backlog – Upravljanje tokom

- Release Flow Management
 - Proces grooming-a treba da bude takav da PB nakon njega bude pripremljen u skladu sa planom tekućeg release-a
 - PB se podeli na tri dela:
 - Must to have – moraju biti u release-u
 - Nice to have – obično idu u sledeći release ako bude prilike
 - Won't have
 - Ovakvo održavanje (delenje) PB pomaže u boljem planiranju release-a



Product Backlog – Upravljanje tokom

- Sprint Flow Management
 - Kada se radi grooming za sprint flow zgodno je PB posmatrati kao cev kroz koju prolaze zahtevi koji ulaze u sprint
 - Grooming postupak traje dok zahtevi „teču“ kroz „cev“
 - Tokom grooming aktivnosti u „cev“ ulaze veći nerazrađeni zahtevi, a iz nje izlaze manji, detlajno razrađeni zahtevi spremni za sprint



Product Backlog – Upravljanje tokom

- Sprint Flow Management
 - Brzina tok zahteva kroz „cev“ mora biti adekvatna
 - Ako je tok (grooming postupak) suviše spor tim neće imati materijala sa kojim može da planira sledeći sprint
 - Ako je tok (grooming postupak) suviše brz može se desiti da je veliki broj zahtev detaljno razrađen, ali usled nekih izmena zahtevi će se morati ponovo razrađivati ili menjati ili čak odbaciti
 - U praksi se pokazalo da je obično dovoljno imati materijala unapred za do 3 sprinta

Product Backlog – Kolik PB-ova imati?

- Obično je pravilo jedan proizvod jedan product backlog
 - Ponekad je teško ustanoviti šta je proizvod
 - Npr. MS Office Suite ili Word, Excel, PowerPoint, ...
- Za velike proizvode često se formira hijerarhijski PB
 - Na vrhu je PB sa epovima
 - Na nižim nivoima su PB-ovi razrađeni epovima, pri čemu se svaki PB odnosi na neku celinu/oblast proizvoda
- Ako više timova radi na proizvodu onda se PB obično strukturira tako da svaki tim vidi samo ono što je relevantno za njega

Procena i brzina

Estimation and Velocity

- Kada se planira upravljanje razvojem nekog proizvoda potreban je odgovor na tri bitna pitanja:
 - Koliko funkcionalnosti će biti kompletirano?
 - Kada će biti završeno?
 - Koliko će koštati?
- U Scrum-u da bi se odgovorilo na ova pitanja potrebno je proceniti veličinu onoga što se razvija i izmeriti brzinu ili tempo sa kojim se posao može završiti
- Na osnovu ovi mera moguće je dati neku procenu o trajanju posla i time cenu posla

Procena i brzina

- Primer procene i brzine

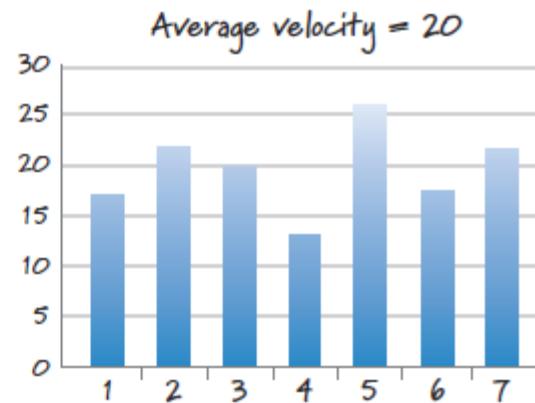
Estimated size + measured velocity = (number of sprints)

Item	Size
Feature A	5
Feature B	3
Feature C	2
Feature D	8
Feature E	2
Feature F	5
Feature G	3
Feature ...	1...
Feature ZX	5
Feature ZY	2
Feature ZZ	1
Feature ...	1...

$$200 \text{ points} \div 20 \text{ points/sprint} = 10 \text{ sprints}$$

$$\sum = 200 \text{ points}$$

Release 1



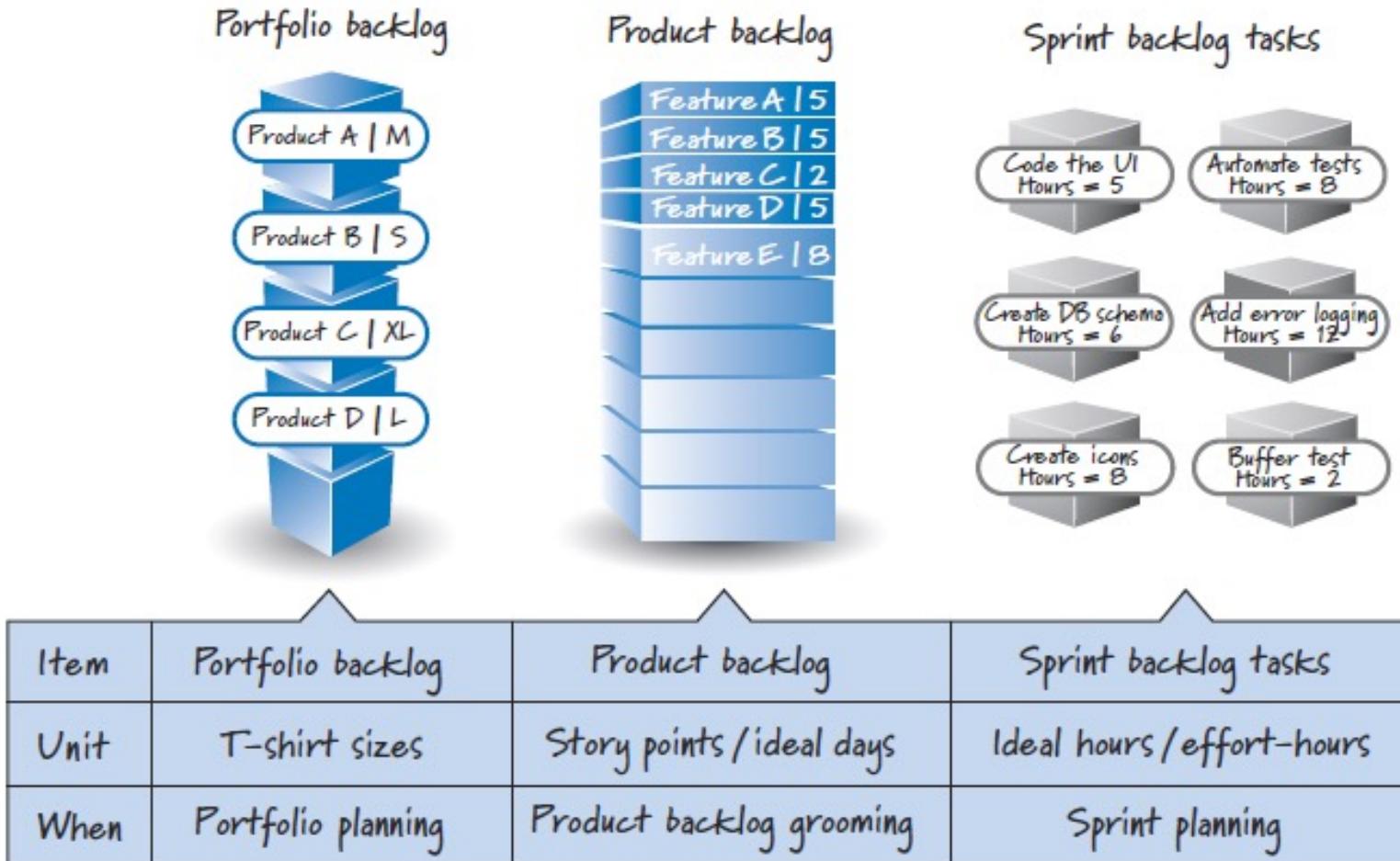
Procena i brzina

- Primer procene i brzine
 - Koliko vremena je potrebno za release 1?
 - Potrebno je izmeriti veličinu release-a 1
 - Veličina se meri tako što se saberu individualne veličine svake stavke product backlog-a koja ide u release 1
 - Kada je poznata veličina potrebno je proceniti brzinu tima, tj. koliko posla tim omično obavi za jedan sprint
 - Na karaju sprinta sumiraju veličine svih stavki PB-a koje su relizovane tokom sprinta – ta suma predstavlja brzinu tima za taj sprint
 - Kad imamo više realizovanih sprintova može se uzet srednja vrednost brzina za svaki sprint
 - Na kraju se broj ukupnih veličina stavki PB-a deli sa prosečnom brzinom za sprint i dobija se broj sprintova potrebnih da se realizuje projekat, tj. dobija se vreme trajanja projekta

Procena i brzina – Šta i kad se procenjuje

- U prethodnom primeru su se za procenu koristili poenu (**story points**)
- Tokom razvoja proizvoda procene obično moraju vršiti na različitim nivoima granularnosti i samim tim koristiti različite jedinice za to
- U većini organizacija procena za potrebe planiranja se vrši na tri različita nivoa detalja
 - Portfolio backlog
 - Product backlog
 - Sprint backlog

Procena i brzina – Šta i kad se procenjuje



Procena i brzina – Šta i kad se procenjuje

- Portfolio Backlog procena
 - Iako Portfolio backlog nije deo standardnog Scrum-a, mnoge organizacije ga poseduju i u njemu se nalazi lista proizvoda/projekata koji se planiraju realizovati
 - Pošto ovaj backlog sadrži globalne stavke sa vrlo malo detalja procena se vrši prilično grubo – obično po veličini majica

Procena i brzina – Šta i kad se procenjuje

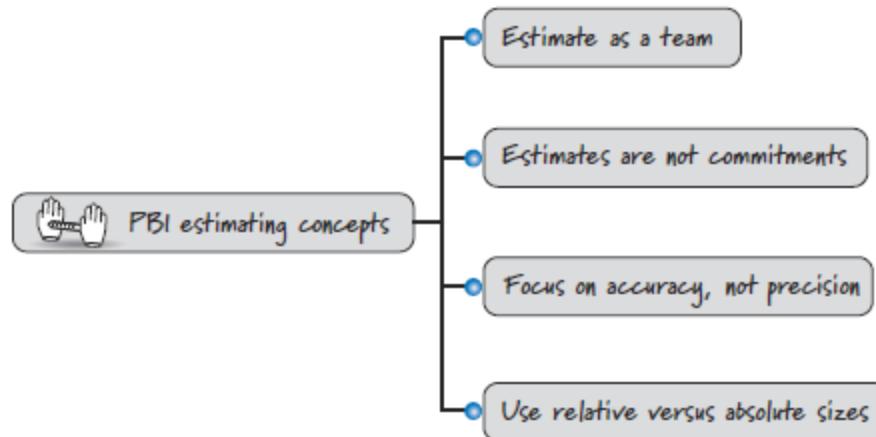
- Product backlog procena
 - Kada stavke PB dođe na red po prioritetu i kada je završen grooming proces kojim su uključeni dodatni detalji za stavke većina timova vrši numeričko procenjivanje u story poenima ili još bolje danima
 - Procena stavki PB se obično vrši u okviru grooming aktivnosti, obično na tzv. „estimation meetings“
 - Neki smatraju da procena stavki PB-a nije neophodna stvar za uspešan Scrum jer s vremenom uigrani timovi mogu bez ocenjivanja da procene koliko stavki PB se može izvršiti u sprintu

Procena i brzina – Šta i kad se procenjuje

- Sprint backlog procena
 - Na najdetaljnijem nivou procenjuju se zadaci (tasks) koji se nalaze u sprint backlog-u
 - Zadaci se obično procenjuju u čovek-satima (ideal hours, effort-hours, man-hours, person-hours)

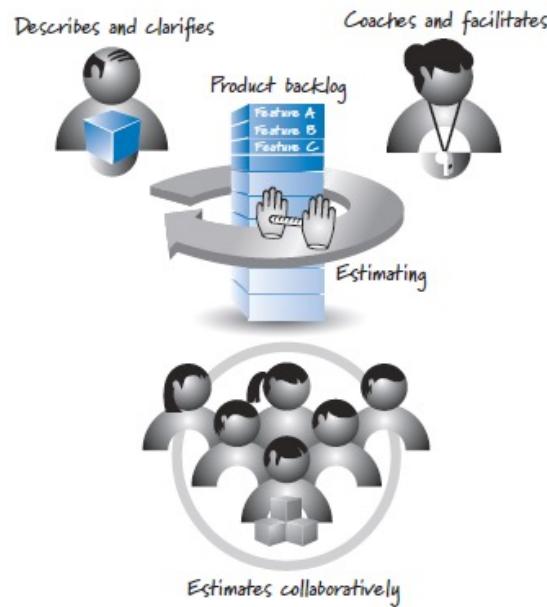
Procena i brzina – koncepti procene PB-a

- Četiri elementa
 - Timska procena
 - Procene nisu i obaveze
 - Fokus je na tačnosti, a ne na preciznosti
 - Korstiti relativne veličine



Procena i brzina – koncepti procene PB-a

- Timska procena
 - Svi članovi razvojnog tima bi trebalo da timski učestvuju u proceni, a ne pojedinci
 - Svaki član tima procenjuje iz svog ugla
 - Product Owner i ScrumMaster su prisutni na proceni, ali oni ne bi trebalo da rade samu procenu



Procena i brzina – koncepti procene PB-a

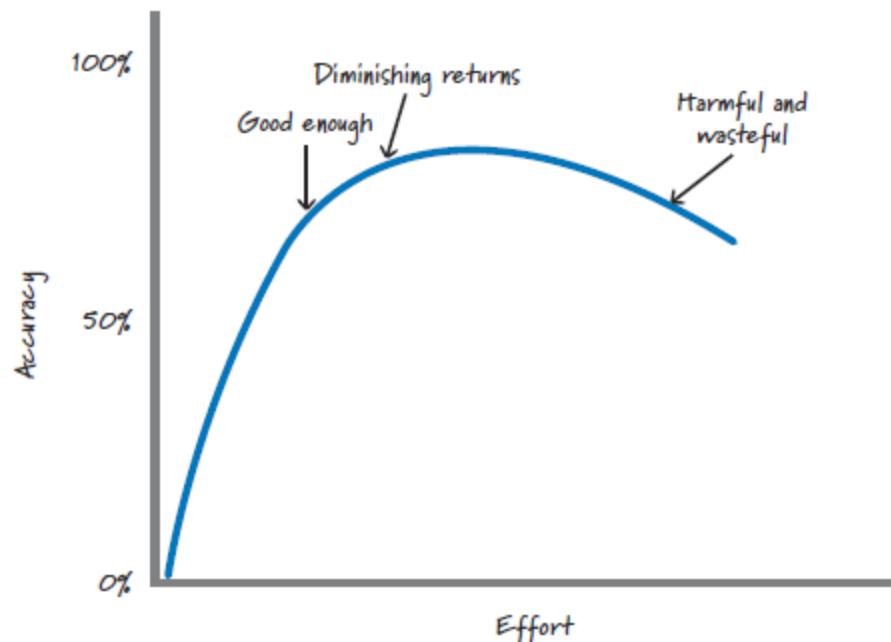
- Procene nisu i obaveze
 - Ovaj koncept se obično tiče menadžera
 - Vrlo često zahtevaju da se tim obaveže na procene - što nije u skladu sa Scrum-om
 - Scrum ne zahteva od razvojnog tima da se obaveže da će sve realizovati u skladu sa navedenim procenama

Procena i brzina – koncepti procene PB-a

- Fokus je na tačnosti, a ne na preciznosti
 - Procene moraju biti tačne, ali ne i preterano precizne
 - Uglavnom preterano precizne procene su obično bespotrebne
 - Utrošeno je dosta truda da bi se takve procene realizovale
 - Sve postaje beskorisno kada se uvidi da se nešto pogrešno razmelo
 - Preporuka je da se utroši truda toliko da procene budu dovoljno dobre

Procena i brzina – koncepti procene PB-a

- Tačnost vs. preciznost



Procena i brzina – koncepti procene PB-a

- Korstiti relativne veličine
 - Preporuka je da se koriste relativne veličine, a ne absolutne
 - Obično poredi koliko se stavke međusobno relativno rezlikuju, a ne absolutno
 - Lakše je proceniti



Procena i brzina – jedinice procene

- Iako Scrum ne definiše standardne jedinice sa procenu stavki PB-a u praksi su najviše zastupljeni story points (cc 70%) i čovek-dani (cc 30%)

Procena i brzina – jedinice procene

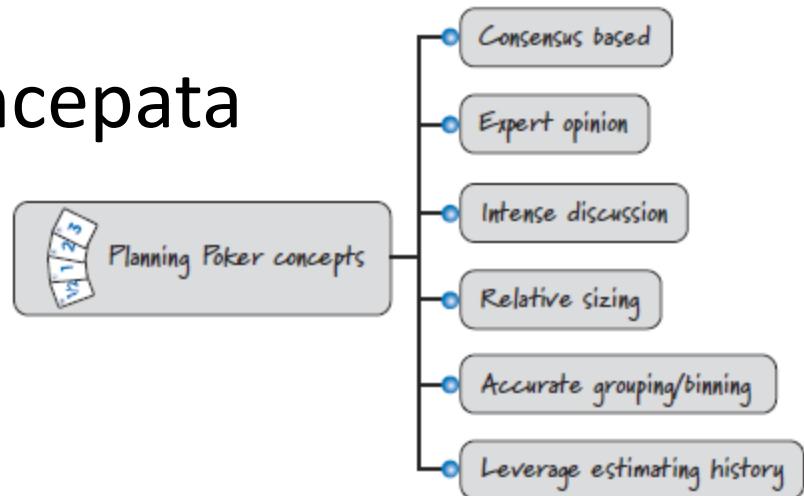
- Story points
 - Sa njima se izražava veličina PB stavki
 - Veličina story points-a je pod uticajem nekoliko faktora uključujući kompleksnost i fizičku veličinu, tj. nekoliko faktora se kombinuje u jednu (relativnu) vrednost
 - Cij je da se različiti story-ji mogu poređiti
 - Npr. $\text{story}_1=2$, $\text{story}_2=8 \rightarrow$ dugi story je 4 puta složeniji od prvog

Procena i brzina – jedinice procene

- Čovek-dani/sati (Ideal days/hours)
 - Broj dana/sati potrebnih da bi se realizovala story
 - Nije isto što i efektivno vreme (elapsed time) kada će nešto biti gotovo
 - Npr. četvrtina u košarci traje 10/12 minuta, a efektivno obično traje dosta duže
 - Ovakva procena nekad zna biti nezgodna jer može uzrokovati pogrešno razumevanje
 - Ako za nešto procenimo da traje 3 čovek-dana ne znači da će to i biti urađeno za 3 dana od danas

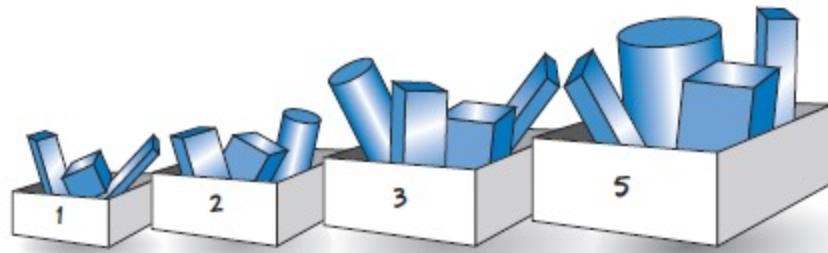
Procena i brzina – Planning Poker

- Planning Poker je tehnika za dimenzionisanje stavki PB-a
- Baziran je na nekoliko koncepata
 - Konsenzus
 - Ekspertsко mišljenje
 - Intenzivna diskusija
 - Relativno dimenzionisanje
 - Precizno grupisanje
 - Iskoristiti prednosti istorije procena



Procena i brzina – Planning Poker

- Skala procene
 - Da bi izveo Planning Poker tim mora da se dogovori oko skale brojeva koju će koristiti za procenu
 - Najčešće korišćena skala je bazirana na malo modifikovanoj sekvenci fibonačijevih brojeva
 - 1, 2, 3, 5, 8, 13, 20, 40 i 100
 - Slične stavke PR se grupišu i dodeljuje im je jedan broj

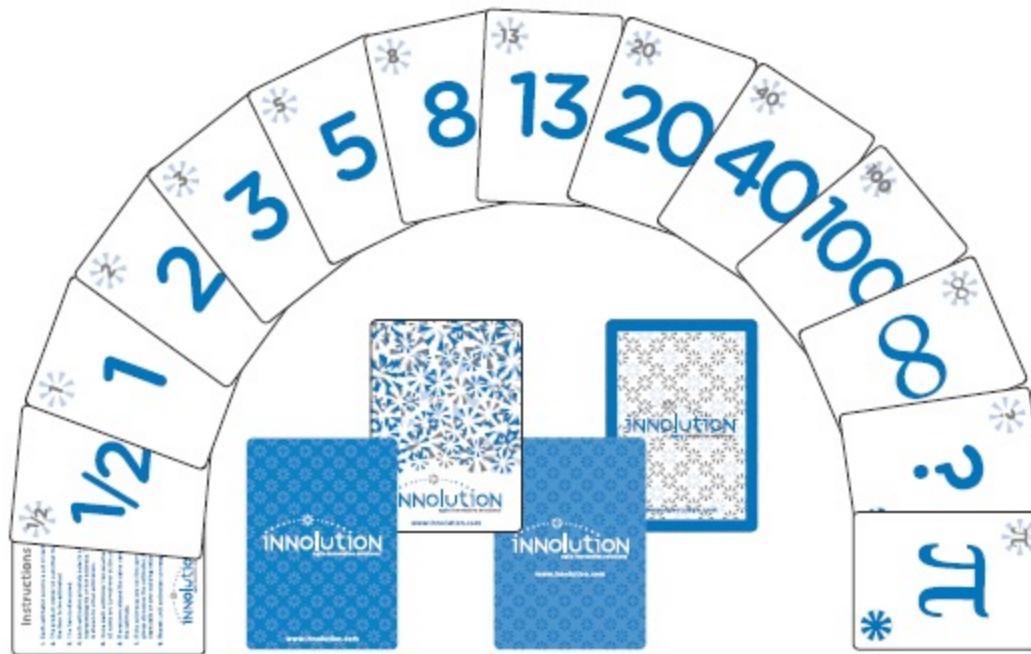


Procena i brzina – Planning Poker

- Kako se igra?
 - Obavezno učestvuje čitav tim
 - Tokom sesije product owner da prezentira, opiše i pojasni sve šta je potrebno
 - ScrumMaster nadgleda sve i treba da pomogne da rezultati budu što kvalitetniji
 - Svaki član razvojnog tima ima šipil Planning Poker karata

Procena i brzina – Planning Poker

- Primer Planning Poker karata



Procena i brzina – Planning Poker

- 0
 - Pojavljuje se u nekim špilovima, označava da je neka stavka ili urađena ili jako mala da nema smisla da joj se dodeljuje vrednost
- $\frac{1}{2}$
 - Koristi se za sitne (minijature) veličine
- 1, 2, 3
 - Koristi se za male stavke
- 5, 8, 13
 - Koristi se za srednje stavke. Za mnoge timove vrednost 13 je gornja granica nečega što ide u sprint, sve preko ove vrednosti bi trebalo da se dekomponuje na više manjih stavki
- 20, 40
 - Koristi se za velike stavke, npr. feature-level ili theme-level stories
- 100
 - Koristi se za jako velike feature ili epic

Procena i brzina – Planning Poker

- ∞ (beskonačno)
 - Stavka je toliko velika da nema smisla da joj se dodeljuje neki broj
- ?
 - Član tima ne razume stavku i zahteva dodatno pojašnjenje od ProductOwner-a. Neki članovi timovi koriste ovu kartu da bi izbegli procenu jer nemaju ideju kakvu procenu da daju
- π (pi)
 - Koristi se kada neki član tim želi da pauzira igru („I'm tired and hungry and I want to get some pie!“). Cesto se koristi i karta sa šoljom kafe

Procena i brzina – Planning Poker

- Pravila igre
 1. Product owner odabira stavku PBI koja se procenjuje i čita je timu
 2. Razvojni tima raspravlja o stavci i eventualno postavlja pitanja product owner-u
 3. Svaki član tima za sebe odabira kartu za tu stavku – to predstavlja njegovu procenu
 4. Kada su svi odabrali kartu za sebe, odabrane karte se pokazuju svima drugima
 5. Ako su svi odabrali istu kartu postignut je konsenzus i to postaje procena za tu PB stavku

Procena i brzina – Planning Poker

- Pravila igre
 6. Ako procene nisu identične počinje diskusija u kojoj se razjašnjava zašto postoji razlika. Obično počinje tako da oni što su dali najviši i najnižu porcenu objašnjavaju svoje procene
 7. Nakon diskusije ide se na korak 3 i sve se ponavlja do se ne dođe do konsenzusa

Procena i brzina – Planning Poker

- U Planning Poker-u cilj nije kompromis u timu već postizanje konsenzusa tima
- Konsenzus se obično postiže kroz dve ili tri runde glasanja
- Pokazalo se da se članovi tima na ovaj način motivišu da razmišljaju o detaljima svake stavke i da tokom igre steknu dodatno znanje o stavici PB-a

Procena i brzina – Šta je brzina?

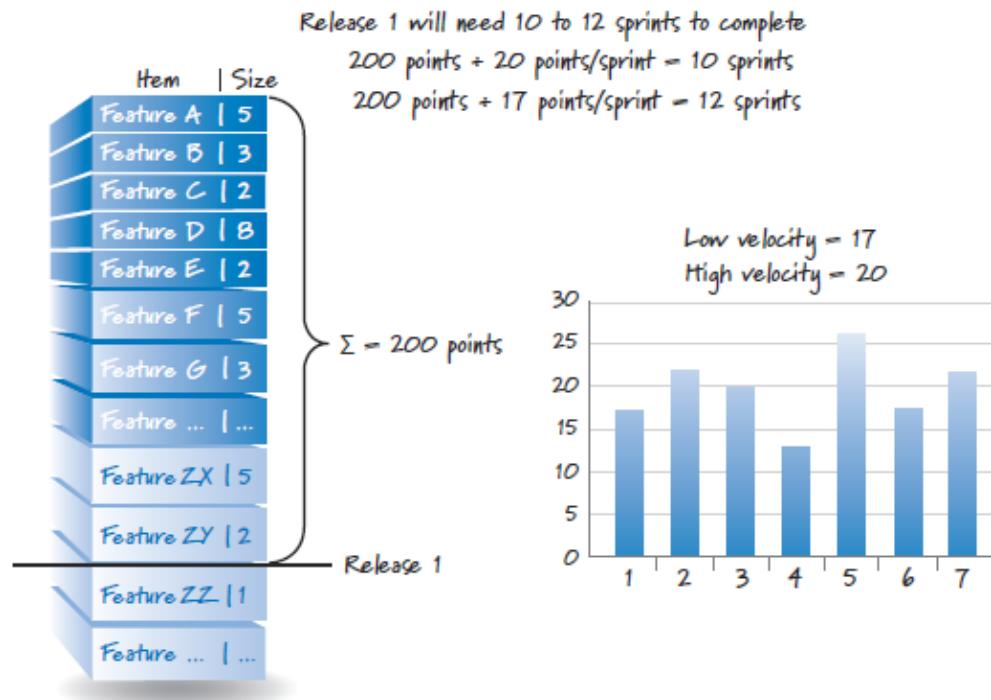
- Brzina je količina posla završena u sprintu
- Meri (računa) se kao suma procena svih stavki koje su završene u sprintu
 - Ne uključuju se stavke koje su parcijalno završene
- Brzina predstavlja osnovu za Scrum planiranje
 - Na osnovu brzine tima može se proceniti koliko sprintova je potrebno za novi release
- Brzina je takođe zgodna kao dijagnostika timu da izvrši evaluaciju koliko dobro primjenjuje Scrum i kako može da poboljša korišćenje Scrum-a

Procena i brzina – Računanje opsega brzina

- Za potrebe planiranja brzina kada se izrazi kao opseg
 - Npr. Tim obično može da uradi 25 do 30 poena u svakom sprintu
- Korišćenje opsega omogućuje nam da budemo tačni, ali ne i preterano precizni
- Pitanja poput „Kada će biti gotovo?“, „Koliko stavki može da se uradi?“, „Koliko će to da košta?“ se obično postavljaju u ranim fazama projekta kada detalji nisu baš poznati i kada tim ima najmanje informacija o projektu
 - U praksi se pokazalo da je opseg brzina zgodan za davanje odgovara na ova i slična pitanja

Procena i brzina – Računanje opsega brzina

- Primer
 - Umesto konkretnih vrednsoti dajemo opseg vrednosti
 - Procenjuje se donja i gornja brzina tima



Tehnički dug

- Pod pojmom tehničkog duga se podrazumevaju prečice kojima namerno idemo kao i mnoge loše stvari koje negativno utiču na softverski sistem
- Tehnički dug uključuje
 - Loš dizajn – dizajn koji je u početku imao smisla sada više nije dobar
 - Defekti – poznati problemi u softveru koji nisu uklonjeni
 - Nedovoljna pokrivenost testovima – postoje delovi softvera koji zahtevaju više testova ali se ne sprovode
 - Preterano manuelno testiranje – manuelno testiranje u segmentima gde bi bolje bilo automatsko
 - Slaba integracija i upravljanje releasovima – sprovođenje ovih aktivnosti na način koji zahteva mnogo vremena i podložno je greškama
 - Slabo iskustvo u platformi/ahrhitekturi
 - ...

Tehnički dug

- Tri vrste
 - Naivni (naive technical debt)
 - Nedovoljna obučenost tima, slabo iskustvo, nedostatak testiranja
 - Može se eliminisati kroz odgovarajuće obuke, bolje razumevanje problema, ...
 - Neizbežan (unavoidable technical debt)
 - Npr. ne možemo predvideti kakav će dizajn biti aktuelan za par godina
 - Obično je nepredvidiv i teško da može da se spreči
 - Strategijski (strategic technical debt)
 - Pogređno donsesene stragijske greške sa nekim softverskim projektom

Tehnički dug - Posledice

- Unpredictable Tipping Point
 - Jedna od karakteristika tehničkog duga je što nepredvidivo raste. U jednom momentu softver dolazi do tačke (tipping point) u kojoj više ne može da se kontroliše; čak i sitne izmene izazivaju velike probleme
- Povećanje vremena za isporuku (Increased Time to Delivery)
 - Što je trenutni dug veći to je brzina budućih radova sporija a time i vreme isporuke novih release-ova raste
- Značajan broj nedostataka (Significant Number of Defects)
 - Razvoj proizvoda postaje složeniji, a time se povećava broj (neotkrivenih)grešaka

Tehnički dug - Posledice

- Rast troškova razvoja i održavanja (Rising Development and Support Costs)
 - Ako je tehnički dug porastao do neke granice čak i male izmene zahtevaju dosta vremena, a time rastu i troškovi realizacije

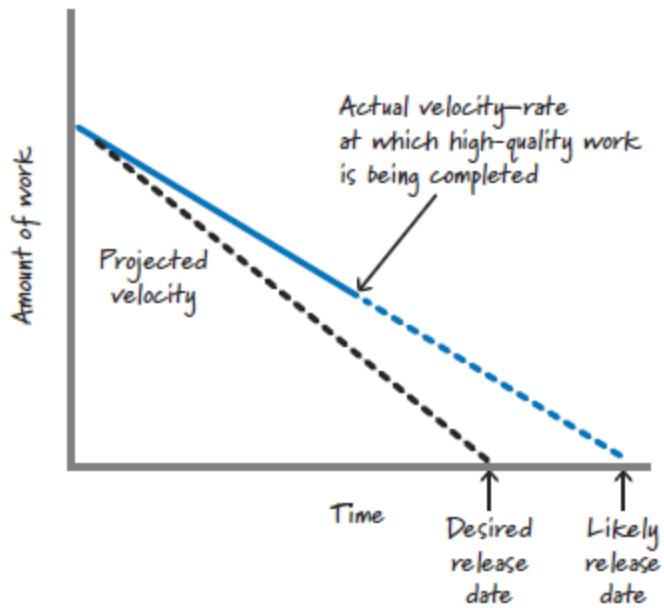


Tehnički dug - Posledice

- Atrofija proizvoda (Product Atrophy)
 - Ako se zbog tehničkog duga prestaju dodavati nove funkcionalnosti i popravka bug-ova proizvod prestaje biti atraktivan tekući i potencijalnim klijentima. Klijenti će pre ili kasnije odustati od tog proizvoda
- Smanjena predvidljivost (Decreased Predictability)
 - Kada postoji značajan tehnički dug teško je vršiti bilo kakva precizna predviđanja; suviše je neodređenosti da bi rezultati bili smisleni
- Slabe performanse (Underperformance)
 - Kako dug raste, ljudi očekuju sve slabije performanse u razvoju i time se smanjuju očekivanja pta je moguće uraditi u softveru
- Razočarenje (Universal Frustration)
 - Obično razvojni tim gubi volju za razvoj/održavanje softvera sa velikim tehničkim dugom što na kraju obično rezultuje napuštanjem tima
- Smanjeno zadovoljstvo klijenata (Decreased Customer Satisfaction)
 - Zadovoljstvo klijenata se smanjuje a frustriranost raste. Vrlo često klijenti ovo sa proizvoda preslikavaju i na razvojni tim i na čitavu kompaniju koja razvoja softver

Tehnički dug - Razlozi

- Pritisak da se zadovolji rok

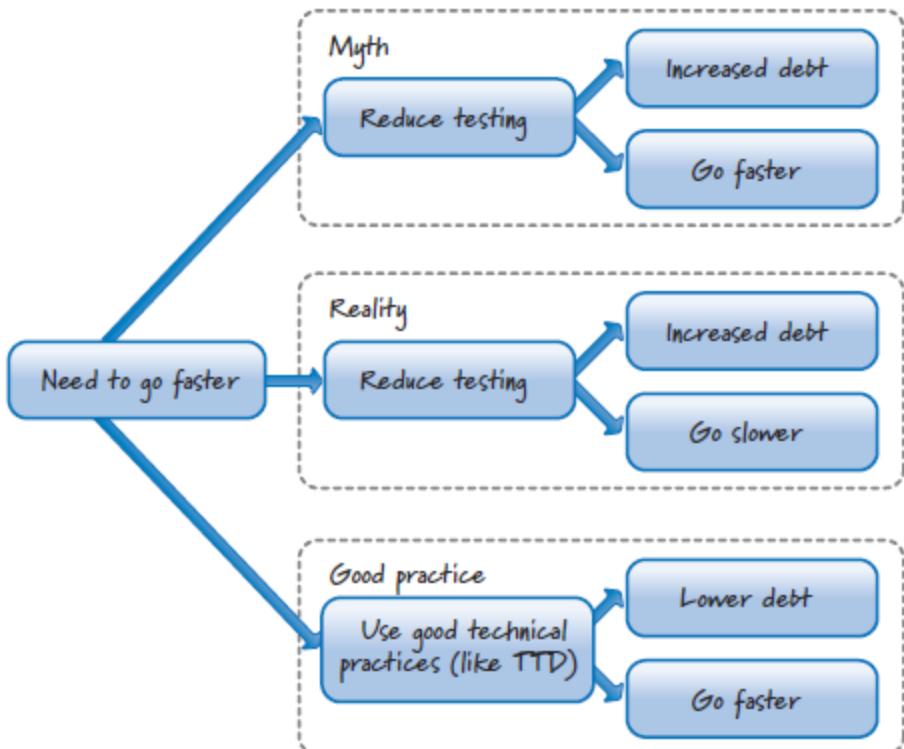
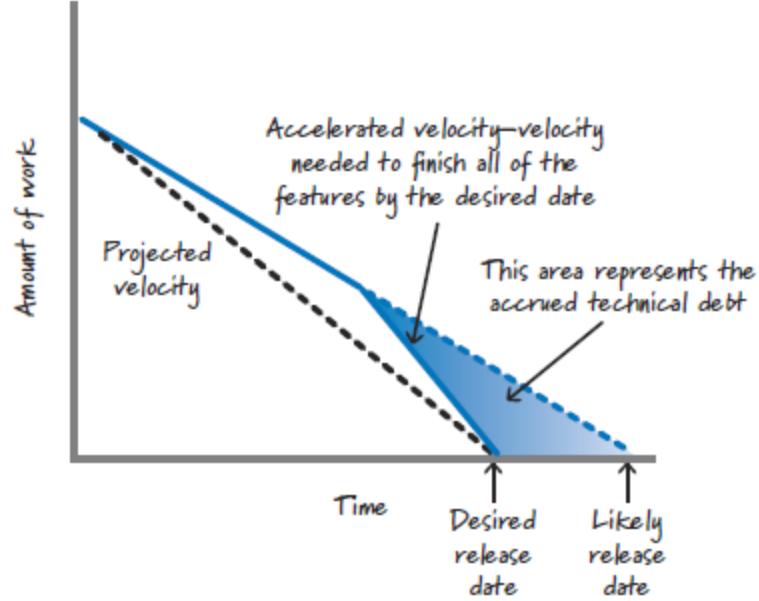


Tehnički dug - Razlozi

- Pokušaj da se lažno poveća brzina razvoja
 - Da li želimo smanjiti zahteve da release bude gotov do željenog datuma ili da se produži rok?
 - Vrlo često se odbijaju obe opcije i od tima se zahtevaju sve funkcionalnosti za kraći rok

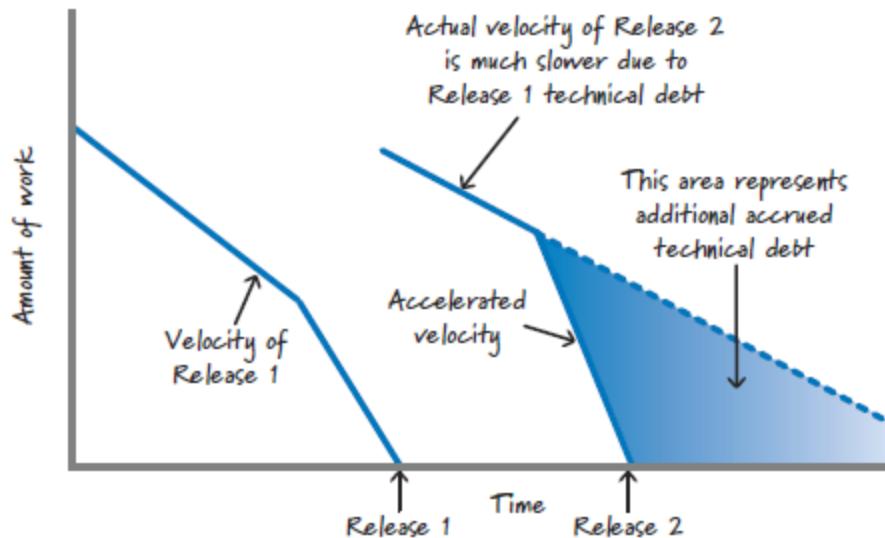
Tehnički dug - Razlozi

- Manje testiranja može ubrzati razvoj



Tehnički dug - Razlozi

- Dug raste na dugu
 - Primer: posledice razvoja release 2 na tehnički dug nastao tokom release-a 1



Tehnički dug – Upravljanje rastom

- U jednom momentu dug se treba preseći
- Obično se prvo zaustavlja naivni tehnički dug
- Takođe, potrebno je uvideti da neizbežnog i strategijskog duga ima u meri koju je moguće sanirati

Tehnički dug – Upravljanje rastom

- Kako kontrolisati?
 - Upravljanje rastom naivnog tehničkog duga
 - Korišćenje dobrih tehničkih iskustava
 - Refaktorisanje koda
 - Korišćenje dobre/čvrste definicije završenosti
 - Na ovaj način se smanjuje ili uopsšte nema tehničkog duga na kraju sprinta
 - Dobro razumevanje ekonomike tehničkog duga
 - Kako tehnički dug utiče na rast troškova

Tehnički dug – Servisiranje

- Poslednja aktivnost u upravljanju tehničkim dugom je servisiranje ili „otplata“ duga
- Obično se geldaju tri kategorije sa stanovišta statusa duga
 - Dug koji se naprsto desio
 - Razvojni tim nije ni bio svesan duga sve do određenog trenutka
 - Poznati tehnički dug
 - Dug koji je poznat timu već neko vreme
 - Ciljani tehnički dug
 - Dug koji je poznat i označen je za servisiranje od strane razvojnog tima

Tehnički dug – Servisiranje

- Na osnovu prethodnih kategorija predložen je sledeći algoritam za servisiranje
 1. Odrediti da li algoritam treba da se servisira, ako treba idese na korak 2.
 2. Ako se tokom kodiranja detektuje dug koji se naprsto desio treba ga popraviti. Ako količina ovog duga prelazi neku razumnu granicu treba ga popraviti do te granice, a ostatak klasifikovati kao poznati tehnički dug
 3. U svakom sprintu određena količina poznatog tehničkog duga treba da se označi u ciljani tehnički dug, tj. da se servisira u okviru tog sprinta

Tehnički dug – Servisiranje

- Nije potrebno servisirati sve tehničke dugove
 - Proizvod koji je pri kraju životnog toka
 - Prototip koji se više neće koristiti
 - Proizvod napravljen da se koristi neki kraći period