



# SCRUM

Februar 2010

*Scrum: Developed and sustained by Ken Schwaber and Jeff Sutherland*

## TAKK TIL:

### *GENERELT*

Scrum er basert på det denne industrien har akseptert som velprøvde "best practices" og som er bevist gjennom årtier. Scrum er videre fundert på empirisk prosessteori. Jim Coplien bemerket en gang til Jeff Sutherland: "Alle kommer til å like Scrum. Når vi har ryggen mot veggen er det vi allerede gjør i dag."

### *MENNESKER*

Blant de tusener som har bidratt til utviklingen av Scrum vil vi gjerne nevne noen av de som var viktige de første ti årene. Først var det Jeff Sutherland som jobbet sammen med Jeff Mckenna, og Ken Schwaber som jobbet sammen med Mike Smith og Chris Martin. Scrum ble formelt introdusert første gang på OOPSLA-konferansen i 1995. I løpet av de neste fem årene bidro Mike Beadle og Martine Devos betydelig. Etter det bidro alle de andre til å raffinere Scrum til det vi kjenner det som i dag.

### *HISTORIKK*

Scrum sin historie kan betegnes som lang i programvareutviklingens verden. For å gjøre ære på de første stedene prosessen ble utprøvd og videreutviklet, nevner vi Individual, Inc., Fidelity Investments og IDX (nå GE Medical).

### *OVERSETTELSE*

Denne guiden har blitt oversatt fra den opprinnelige engelske versjonen skrevet av Ken Schwaber og Jeff Sutherland. Bidragsyttere til denne oversettelsen er Mari Wien, Nina Hanssen og Simen Fure Jørgensen fra Iterate AS.

## HENSIKT

Scrum har blitt brukt for å utvikle komplekse produkter siden tidlig på nittitallet. Denne guiden beskriver hvordan man kan benytte Scrum til å bygge produkter. Scrum er ikke en prosess eller en teknikk for å bygge produkter, det er snarere et rammeverk for hvordan du kan benytte deg av ulike prosesser og teknikker. Rollen til Scrum er å få frem den relative effektiviteten til ditt utviklingsløp. **På den måten kan du forbedre utviklingen, mens Scrum gir det et rammeverk for å utvikle komplekse produkter.**

## SCRUM-TEORI

Scrum, som er fundert i empirisk teori for prosesskontroll, benytter en iterativ og inkrementell tilnærming for å optimalisere forutsigbarhet og kontrollere risiko. Tre pilarer støtter oppunder alle implementeringer av empirisk prosesskontroll.

## DEN FØRSTE PILAREN ER GJENNOMSIKTIGHET

Gjennomsiktighet sikrer at de aspektene ved en prosess som påvirker utfallet er synlige for de som styrer prosessen. Ikke bare må disse aspektene være gjennomsiktige, men det som vises må også være kjent. Det betyr at når noen som inspiserer en prosess mener at noe er ferdig må det tilsvare deres definisjon av "ferdig".

## DEN ANDRE PILAREN ER INSPEKSJON

De forskjellige aspektene av en prosess må inspiseres ofte nok til at uakseptable variasjoner i prosessen kan oppdages. Frekvensen av inspeksjonene må ta hensyn til at alle prosesser endres av selve inspeksjonen. Et problem oppstår når den nødvendige frekvensen av inspeksjoner overstiger toleransen for inspeksjoner i prosessen. Heldigvis ser ikke dette ut til å gjelde for softwareutvikling. Den andre faktoren er kunnskapen og nøyaktigheten til menneskene som inspiserer resultatene av arbeidet.

## DEN TREDJE PILAREN ER TILPASNING

Hvis inspektøren etter inspeksjon mener at et eller flere aspekter ved prosessen er utenfor de akseptable grensene, og at produktet på grunn av dette vil bli underkjent, må inspektøren justere prosessen eller produktet som produseres. Tilpasningen må gjøres så raskt som mulig for å minimere videre avvik.

Det er tre punkter for inspeksjon og tilpasning i Scrum. Det daglige Scrum-møtet brukes for å inspisere fremdriften mot sprintmålet, og for å gjøre tilpasninger som optimaliserer verdien av den neste arbeidsdagen. I tillegg brukes sprintgjennomgang og sprintplanleggingsmøtet til å inspisere fremdriften mot release-målet, og til å gjøre tilpasninger som optimerer verdien av den neste sprinten. Til slutt brukes sprint-retrospektivet til å belyse den forrige sprinten og til å avgjøre hvilke tilpasninger som vil gjøre den neste sprinten mer produktiv og givende.

## SCRUMS INNHOLD

Scrum-rammeverket består av et sett av Scrum-team og deres assosierte roller, tidsbokser, artifakter og regler.

Scrum-team er bygget for å optimalisere fleksibilitet og produktivitet. For å oppnå dette er de selv-organiserende, kryss-funksjonelle og jobber i iterasjoner. Hvert Scrum-team har tre roller:

1. Scrum-masteren, som er ansvarlig for at prosessen er forstått og etterfulgt.
2. Produkteieren, som er ansvarlig for å maksimere verdien av det arbeidet Scrum-teamet gjør.
3. Teamet, som utfører arbeidet.

Teamet består av utviklere med alle nødvendige ferdigheter for å gjøre produkteierens krav om til en potensielt leveranseklar del av produktet ved slutten av sprinten.

Scrum bruker tidsbokser for å skape forutsigbarhet. Elementer av Scrum hvor tidsboksene benyttes er **release-planleggingsmøtet**,

**sprintplanleggingsmøtet**, det **daglige Scrum-møtet**, **sprintgjennomgangen** og **sprintretrospektivet**. Hjertet i Scrum er **sprinten**. Sprinten er en iterasjon som er en måned eller kortere av lengde og har samme lengde gjennom hele utviklingsløpet. Alle sprinter bruker det samme Scrum-rammeverket, og alle sprinter leverer et inkrement av et produkt som potensielt kan leveres. En sprint starter umiddelbart etter den forrige.

Scrum bruker fire artifakter. Produktkøen er en prioritert liste over alt som det kan være behov for i produktet. Sprintkøen er en liste av oppgaver hentet fra produktkøen, og som gjør én sprint om til et inkrement av et potensielt leverbart produkt. En *release-burndown* er et mål på gjenværende produktkø over tid. En *sprint-burndown* måler den gjenværende andelen av sprintkøen gjennom forløpet av en sprint.

**Regler** binder sammen Scrums tidsbokser, roller og artifakter.

Reglene er beskrevet i dette dokumentet. For eksempel er det en regel i Scrum at bare team-medlemmer - altså menneskene som er forpliktet til å gjøre produktkøen om til et inkrement - kan snakke når det avholdes daglig Scrum-møte. Ulike måter å implementere Scrum på, og som er forslag snarere enn regler, er beskrevet i tips-bokser.

#### Tips

Når prosessmetodikken ennå ikke er uttalt, er det forventet at Scrum-brukerne likevel vet hva de skal gjøre. Ikke bruk tid på å lage en perfekt metode, fordi utfordringene vil forandre seg fortløpende. Prøv i stedet noe ut, og se og lær hvordan det fungerer. Inspiser-og-tilpass mekanismene innebygd i Scrum vil veilede deg videre.

## ROLLER I SCRUM

Scrum-teamet består av Scrum-masteren, produkteieren og teamet. Scrum-teamet kalles "griser". Alle andre er "høns". Høns kan ikke fortelle grisene hvordan de skal arbeide. Disse uttrykkene kommer fra følgende historie:

*"En høne og en gris går sammen når hønen sier: "La oss starte en restaurant!"*

*Grisen tenker på det og spør: "Hva skulle vi ha kalt denne restauranten?"*

*Hønen svarer "Egg og bacon!".  
Grisen sier: "Nei takk. Jeg ville ha vært forpliktet, mens du ville bare  
ha vært involvert!"*

## SCRUM-MASTEREN

Scrum-masteren er ansvarlig for å forsikre seg om at Scrum-teamet følger verdiene, praksisene og reglene i Scrum. Scrum-masteren skal hjelpe Scrum-teamet og organisasjonen med å adoptere Scrum. Scrum-masteren lærer opp teamet ved hjelp av coaching og ved å lede det til å bli mer produktivt og til å lage produkter av høyere kvalitet. Scrum-masteren hjelper Scrum-teamet med å forstå og bruke egen-ledelse og kryss-funksjonalitet. Scrum-masteren hjelper også Scrum-teamet til å gjøre en best mulig jobb i en organisasjon som muligens enda ikke er optimalisert for kompleks produktutvikling. Når Scrum-masteren hjelper til med disse endringene, kalles det å fjerne hindringer. Men Scrum-masteren leder altså ikke Scrum-teamet, det organiserer seg selv.

### Tips

Scrum Masteren arbeider sammen med kunden og ledelsen om å identifisere og utpeke en produkteier. Scrum Masteren forklarer for og lærer opp produkteieren. Det er forventet at produkteieren er i stand til å optimalisere målene for verdiskapning med Scrum. Hvis ikke, er det Scrum Masteren som holdes ansvarlig.

### Tips

En Scrum Master kan være en av team-medlemmene; for eksempel en utvikler som utfører sprint-oppgaver. Det viser seg likevel at det ofte leder til konflikter når Scrum Masteren må velge mellom å fjerne hindringer og utføre team-oppgaver. Scrum Masteren skal derfor aldri være produkteieren.

## PRODUKTEIEREN

Produkteieren er den ene personen som er ansvarlig for å holde oversikt over produktkøen og for å sikre verdien av det arbeidet teamet utfører. Denne personen vedlikeholder produktkøen og forsikrer seg om at den er tilgjengelig for alle. Alle vet hvilke punkter som har høyest prioritet, slik at alle vet hva som vil bli jobbet med.

### Tips

I kommersiell programvareutvikling kan produkteieren være den produktansvarlige. I bedriftsinterne utviklingsprosjekter kan produkteier være den som er ansvarlig for forretningsutviklingen for dette produktet.

Produkteieren er en person, ikke en gruppe. Det kan finnes grupper som gir råd eller som utøver innflytelse over denne personen, men individer som ønsker å endre prioriteten på noe må overbevise produkteieren. Organisasjoner som innfører Scrum kan finne at det innvirker på deres metoder for å sette prioriteringer og krav over tid.

### Tips

Produkteier kan være en del av Scrum-teamet, som for eksempel gjør utvikling. Men dette kan også påvirke negativt produkteiers evne til å samarbeide med stakeholders. Derfor kan produkteier aldri være Scrum Master.

For at produkteieren skal lykkes må alle i organisasjonen ha respekt for hans eller hennes beslutninger. Ingen har myndighet til å be teamet om å jobbe etter andre prioriteringer, og teamet skal heller ikke høre på noen som har andre oppfatninger. Produkteierens beslutninger er synlige gjennom innholdet og prioriteringene i produktkøen. Denne synligheten fordrer at produkteieren gjør sitt beste, og det gjør at rollen som produkteier både er krevende og givende.

## TEAMET

Team av utviklere gjør produktkøen om til inkremitter av potensielt leverbar funksjonalitet i hver Sprint. Teamene er også kryss-funksjonelle. Team-medlemmene må ha alle de nødvendige ferdighetene for å skape et inkrement.

Team-medlemmer har ofte spesialiserte ferdigheter, som programmering, kvalitetskontroll, forretningsanalyse, arkitektur og design av brukergrensesnitt eller databaser. Uansett så ser ferdighetene som Team-medlemmene deler, det vil si evnen til å identifisere et krav og til å gjøre det om til et brukbart produkt, ut til å være viktigere enn de ferdighetene de ikke har. Folk som nekter å kode fordi de er arkitekter eller designere er ikke gode team-medlemmer. Alle stiller opp, selv om det innebærer å lære noe nytt eller å gjenoppfriske gamle kunnskaper. Det er ingen titler i teamet, og til denne regelen er det ingen unntak. Team skal heller ikke ha undergrupper som er dedikert til spesielle områder som testing eller forretningsanalyse.

Teamene er selvorganiserende. Ingen, selv ikke Scrum-masteren, kan fortelle teamet hvordan det skal gjøre produktkøen om til inkremitter av ny funksjonalitet. Teamet skal finne ut dette selv. Hvert team-medlem tilfører sin ekspertise til alle problemene. Synergien som dette resulterer i forbedrer hele teamets generelle effektivitet.

Den optimale størrelsen på et team er syv personer, pluss eller minus to. Når det er færre enn fem team-medlemmer er det mindre interaksjon og som et resultat av det mindre økning i produktiviteten. Videre kan teamet møte på problemer med manglende ferdigheter i deler av sprinten og ikke være i stand til å levere et produkt som er ferdig til leveranse. Er det flere enn ni medlemmer blir det rett og slett behov for mye koordinering. Store team genererer for mye kompleksitet til at en empirisk prosess kan håndtere det. Vi har imidlertid støtt på suksessfulle team som har overskredet øvre eller nedre grense når det gjelder ønsket størrelse. Produkteieren og Scrum-masteren er ikke inkludert i dette tallet, hvis de ikke også er "griser".

Sammensetningen av teamet kan endres ved slutten av en sprint. Hver gang teamet endres minsker produktiviteten som er oppnådd gjennom selv-organiseringen. Det er derfor nødvendig å ta hensyn til dette når man endrer team-sammensetningen.

## TIDSBOKSER

Tidsboksene i Scrum er **release-planleggingsmøtet, sprinten, sprintplanleggingsmøtet, sprintretrospektivet** og det **daglige Scrum-møtet**.

## RELEASE-PLANLEGGINGSMØTET

Hensikten med release-planlegging er å etablere en plan og mål som Scrum-teamene og resten av organisasjonen kan forstå og kommunisere. Release-planleggingen gir svar på følgende spørsmål: "Hvordan kan gjøre visjonen om til et vinnende produkt på best mulig måte?". "Hvordan kan vi tilfredsstillere eller overgå den forventede kundetilfredshet og lønnsomhet?". Release-planen fastsetter målene, elementene i produktkøen med høyest prioritet, de store risikoene og hvilken høynivå-funksjonalitet en leveranse vil inneholde. Den etablerer også en sannsynlig leveransedato og total kostnad som bør stemme godt dersom ingen forutsetninger endrer seg. Organisasjonen kan så inspisere fremdriften og gjøre endringer på release-planen etter hver sprint.

Release-planlegging er valgfritt. Hvis Scrum-teamet starter sitt arbeid uten dette møtet vil fraværet av møtets artifakter vise seg som en hindring som må fjernes. Arbeid for å fjerne denne hindringen vil bli en oppgave i produktkøen.

Produkter som er bygget iterativt med Scrum, hvor hver sprint utgjør et inkrement av et produkt, starter med det mest verdifulle og det mest risikable. Ytterligere sprinter tilføyer flere inkremitter av produktet. Hvert inkrement er en potensielt leverbar del av hele produktet. Når nok inkremitter har blitt tilføyd, slik at produktet har verdi og er av nytte for interessentene kan produktet leveres.

De fleste organisasjoner har allerede en release-prosess. I de fleste av disse prosessene blir mesteparten av planleggingen utført ved oppstart og forblir uendret etterhvert som tiden går. Med release-planlegging i Scrum defineres overordnede mål og sannsynlige utfall. Denne planleggingen kan utføres på 15-20% av den tiden organisasjoner vanligvis benytter på å lage en tradisjonell release-plan. En Scrum-release gjennomfører *just-in-time* planlegging ved hver sprintgjennomgang og sprintplanleggingsmøte, og i tillegg *just-in-time* planlegging på det daglige Scrum-møtet. Tilsammen utgjør

planleggingsarbeidet i Scrum litt mer innsats enn tradisjonell release-planlegging.

Release-planlegging krever estimering og prioritering av produktkøen. Det er mange teknikker for å gjøre dette som ligger utenfor Scrum, men som allikevel er av nytte når det brukes sammen med Scrum.

## SPRINTEN

En sprint er en iterasjon. Sprinter har tidsbokser. Underveis i sprinter sørger Scrum-masteren for at ingen endringer iverksettes som påvirker sprintmålet. Både team-sammensetning og kvalitetsmål forblir konstante gjennom en sprint. Sprinten består av sprint-planleggingsmøtet, utviklingsarbeidet, sprintgjennomgangen og sprintretrospektivet. Sprinter skjer etter hverandre uten opphold mellom sprintene.

Et prosjekt benyttes til å oppnå noe. I programvareutvikling brukes et prosjekt til å lage et produkt eller et system. Ethvert prosjekt består av en definisjon av det som skal lages, en plan for å lage det, arbeidet som utføres i henhold til planen og det resulterende produktet. Ethvert prosjekt har en horisont, altså en tidsramme for når planen er gyldig. Hvis horisonten er for lang kan prosjektdefinisjonen ha endret seg, for mange variable kan ha kommet til, risikoen kan være for stor og så videre. Scrum er et rammeverk for et prosjekt hvor horisonten ikke er mer enn en måned lang og hvor det er nok kompleksitet for at en lengre horisont er risikabelt. Prosjektets forutsigbarhet må kontrolleres i alle fall månedlig, slik at risikoen for at prosjektet mister kontrollen eller blir uforutsigbar blir begrenset i alle fall en gang i måneden.

### Tips

Hvis teamet fornemmer at det har forpliktet seg til for mye, gjennomføres det et møte med produkteier for å ta ned scopet fra produktbackloggen i inneværende sprint. Hvis teamet mener det har tid til overs, kan det i samarbeid med produkteier tas inn nye oppgaver fra produktbackloggen.

Sprinter kan avlyses før en sprint-tidsboks er over. Bare produkteieren kan avlyse en sprint, men han eller hun kan gjøre dette etter påvirkning fra interessentene, teamet eller Scrum-masteren. Hva slags forhold tilsier at sprinten må avlyses?

### Tips

Når Scrum-teamet starter opp, kan to-ukers sprinter sørge for at teamet ikke famler. Sprinter av denne lengden kan synkroniseres og slås sammen med et annet Scrum-teams inkrement.

Ledelsen kan ha behov for å avlyse en sprint dersom målet ikke lenger er gyldig. Dette kan inntreffe hvis organisasjonen endrer retning eller hvis markedsmessige eller teknologiske forhold endrer seg. Generelt sett bør man avlyse en sprint dersom det ikke lenger gir mening etter forholdene. Det er verdt å merke seg at ettersom sprintene er korte er det sjelden aktuelt å avlyse en sprint.

Når en sprint avlyses gjennomgås ferdigstilte oppgaver i produktkøen. De blir akseptert hvis de utgjør et potensielt leverbart inkrement. Alle andre krav i produktkøen legges tilbake med sine opprinnelig estimater. Eventuelt utført arbeid på disse kravene anses som å være tapt. Avlysning av sprinter forbruker ressurser ettersom alle må omgruppere i et nytt sprintplanleggingsmøte for å planlegge en ny sprint. Sprintavbrudd er ofte traumatiske for teamet og er svært uvanlige.

## SPRINTPLANLEGGINGSMØTET

Iterasjonen blir planlagt i sprintplanleggingsmøtet. Møtet er tidsbokset til åtte timer for en sprint på én måned. For kortere sprinter allokeres omtrent 5% av den totale sprintlengden til dette møtet. Møtet består av to deler. Den første delen, en fire timers tidsboks, omhandler hva som skal gjennomføres i den kommende sprinten. Den andre delen, en ny fire timers tidsboks, omhandler hvordan teamet skal bygge denne funksjonaliteten inn i et inkrement av produktet i løpet av sprinten.

De to delene av møtet utgjør derfor "hva?" og "hvordan?". Noen Scrum-team kombinerer disse to delene. I den første delen tar teamet for seg spørsmålet "hva?". Her presenterer produkteieren den topp-prioriterte delen av produktkøen. Sammen finner man hvilken funksjonalitet som skal utvikles i løpet av den kommende sprinten. Inn i dette møtet tar man med seg produktkøen, den forrige leveransen, teamets kapasitet og teamets tidligere ytelse. Den andelen av produktkøen teamet velger å utvikle er bare teamets avgjørelse. Bare teamet kan vurdere hva det kan oppnå i løpet av den kommende sprinten.

Etter å ha valgt en andel av produktkøen utformes sprintmålet. Sprintmålet er en målsetning som vil bli oppnådd gjennom implementeringen av den valgte andelen av produktkøen. Målsetningen gir teamet veiledning når det gjelder hensikten med et inkrement. Sprintmålet er et subsett av release-målet.

Årsaken til at man har et sprint-mål er å gi teamet litt rom med tanke på funksjonaliteten. Et eksempel på mål for en sprint kan være: "Å automatisere funksjonaliteten for å modifisere kundekontoer gjennom å introdusere en sikker og reverserbar transaksjonsmekanisme". Underveis i sprinten har teamet dette målet i bakhodet. For å tilfredsstillende målet implementeres funksjonaliteten med den tiltenkte teknologien. Hvis oppgaven viser seg å være vanskeligere enn det teamet trodde, så kan teamet samarbeide med produkteieren og bare delvis implementere funksjonaliteten.

I den andre delen av sprintplanleggingsmøtet svarer teamet på spørsmålet om "Hvordan?". I løpet av de fire siste timene av sprintplanleggingsmøtet finner teamet ut hvordan det skal gjøre den valgte andelen av produktkøen ("Hva?") om til et ferdigstilt inkrement. Teamet starter vanligvis med å designe arbeidet. Underveis i designet identifiserer teamet oppgaver. Disse detaljerte oppgavene er det som trengs for å konverte produktkøen for denne sprinten om til fungerende kode. Oppgavene bør være på et slikt nivå at de ikke tar mer enn en dag å fullføre. Denne listen over oppgaver kalles sprintkøen. Teamet selvorganiserer seg slik at oppgavene i sprinten tildeles og arbeidet kan begynne, enten i sprintplanleggingsmøtet eller *just-in-time* i selve sprinten.

#### Tips

Vanligvis vil bare 60-70% av den totale sprint-backloggen gåes nøye i gjennom i sprint-planleggingen. Resten tas senere eller gis grove estimater, og som deles opp senere i sprinten.

Produkteieren er tilstede i den andre delen av sprintplanleggingsmøtet for å avklare spørsmål rundt produktkøen og for å hjelpe teamet med å gjøre avveininger. Hvis teamet mener at det har for mye eller for lite å gjøre kan det revurdere produktkøen sammen med produkteieren. Teamet kan også invitere andre til å delta i møtet for å tilføre teknisk eller forretningsmessig ekspertise. Et nytt team forstår som regel for første gang at det vil lykkes eller feile *sammen*, ikke individuelt, i dette møtet. Teamet innser at det må stole på seg selv. Det starter å organisere seg selv, slik at det etterhvert får karakteristikkene til et ekte team.

## SPRINTGJENNOMGANG

Ved avslutningen av sprinten avholdes det et møte for sprintgjennomgang. Dette er et møte som er tidsbokset til fire timer for en månedslang sprint. For sprinter av kortere varighet må dette møtet ikke ta mer enn 5% av den totale sprintlengden. I sprintgjennomgangen diskuterer Scrum-teamet og interessentene det som nettopp er fullført. Basert på dette, og på endringer som har kommet i produktkøen underveis i sprinten, finner de ut hva det neste som skal gjøres. Dette er et uformelt møte hvor presentasjon av funksjonalitet skal fremme diskusjonen om de neste skrittene.

Møtet inkluderer i alle fall disse elementene: Produkteieren identifiserer hva som har blitt gjort og hva som ikke har blitt gjort. Teamet diskuterer hva som gikk bra i sprinten, hvilke problemer som oppsto og hvordan disse ble løst. Teamet demonstrerer så resultatet av det arbeidet som ble gjort og svarer på spørsmål. Produkteieren presenterer så produktkøen slik den nå står. Han eller hun anslår sannsynlige leveransedatoer med ulike antagelser om hastighet. Hele gruppen diskuterer så det som er gjennomgått og hva dette betyr i forhold til de neste skrittene. Sprintgjennomgangen gir verdifull informasjon til det kommende sprintplanleggingsmøtet.

## SPRINTRETROSPEKTIVET

Etter sprintgjennomgangen og før sprintplanleggingsmøtet avholder Scrum-teamet et sprintretrospektiv. I løpet av denne tre timer lange tidsboksen oppfordrer Scrum-masteren teamet til å revidere, innenfor Scrums prosessrammeverk og praksiser, sin utviklingsprosess for å gjøre den mer effektiv og givende i neste sprint. Det finnes mange bøker som beskriver teknikker som er gode å bruke i retrospektiver.

Hensikten med retrospektiver er å inspisere hvordan den forrige sprinten fungerte med tanke på mennesker, relasjoner, prosesser og verktøy. Denne inspeksjonen skal identifisere og prioritere de viktigste elementene som fungerte bra og de elementene som kunne ha gjort det enda bedre om det hadde vært gjort annerledes. Dette inkluderer sammensetningen av Scrum-teamet, møter, verktøy, definisjonen av ferdig, kommunikasjonsmetoder og prosessen for å gjøre krav fra produktkøen om til noe som er ferdig. Ved avslutningen av sprintretrospektivet skal Scrum-teamet ha identifisert utførbare forbedringstiltak som det implementerer i løpet av den neste sprinten. Disse endringene utgjør tilpasningene etter den empiriske inspeksjonen.

## DET DAGLIGE SCRUM-MØTET

Hvert team møtes daglig for et 15 minutters møte for å inspisere og tilpasse prosessen. Dette er det daglig Scrum-møtet. Dette møtet finner sted på samme tid og sted gjennom hele sprinten. I løpet av møtet forklarer hvert team-medlem følgende:

1. Hva han eller hun har utrettet siden forrige møte.
2. Hva han eller hun skal gjøre før neste møte.
3. Hvilke hindringer han eller hun ser på veien.

Daglige Scrum-møter forbedrer kommunikasjonen, eliminerer andre møter, identifiserer og fjerner hindringer på veien, belyser og fremmer raske beslutninger og forbedrer alles kunnskapsnivå i prosjektet.

Scrum-masteren sørger for at teamet avholder møtet. Teamet er ansvarlig for utførelsen av møtet. Scrum-masteren lærer teamet å holde møtet kort ved å håndheve reglene og sørge for at team-medlemmene er kortfattede. Scrum-masteren håndhever også regelen om at "høns" ikke har lov til å snakke eller på noen måte forstyrre det daglige Scrum-møtet.

Det daglige Scrum-møtet er ikke et statusmøte. Det er ikke for noen andre enn de som gjør den valgte andelen av produktkøen om til et inkrement, altså teamet. Teamet har forpliktet seg til å nå et sprintmål og til å implementere en andel av produktkøen. Det daglige Scrum-møtet er en inspeksjon av fremdriften mot sprintmålet (de tre spørsmålene). Oppfølgingsmøter skjer vanligvis for å gjøre tilpasninger på det kommende arbeidet i sprinten. Hensikten er å optimere sannsynligheten for at teamet når sitt mål. Dette er et nøkkelmøte for å inspisere og tilpasse den empiriske Scrum-prosessen.

## SCRUM-ARTIFAKTER

Scrum-artifakter inkluderer produktkøen, burndown-diagrammet for en release, sprintkøen og burndown-diagrammet for en sprint.

## PRODUKTKØ OG RELEASE-BURNDOWN

Kravene til produktene som teamet eller teamene utvikler er listet i produktkøen. Produkteieren er ansvarlig for produktkøen, dens innhold, tilgjengelighet og prioritering. Produktkøen er aldri komplett.

Det første forsøket på å skrive den utgjør bare de i utgangspunktet kjente og best forståtte kravene. Produktkøen utvikler seg etterhvert som produktet og bruksområdene for produktet utvikler seg. Backloggen er dynamisk i og med at den kontinuerlig endrer seg for å identifisere hvilke egenskaper produktet må ha for å være egnet, konkurransedyktig og brukbart. Så lenge produktet eksisterer, eksisterer også produktkøen.

Produktkøen representerer alt som er nødvendig for å utvikle og lansere et vellykket produkt. Det er en liste over alle egenskaper, funksjoner, teknologier, forbedringer og feilrettinger

som utgjør endringene som vil gjøres på produktet før fremtidige leveranser. Kravene i produktkøen har beskrivelse, prioritet og estimat. Prioritet er drevet av risiko, verdi og nødvendighet. Det finnes mange teknikker for å vurdere disse tingene.

Produktkøen er sortert etter prioritet. Den høyest prioriterte delen av produktkøen bestemmer de første utviklingsaktivitetene. Jo høyere prioriteten er, jo mer haster det, jo mer gjennomtenkt er det og jo større enighet er det om verdien. Høyt prioriterte krav er klarere og har mer detaljert informasjon enn de lavt prioriterte. Med dette følger også bedre estimater. Jo lavere prioritet, jo mindre detalj er det, helt inntil oppgaven er diffus.

Når et produkt er i bruk, når verdien av det øker og når markedet gir tilbakemelding på det blir produktkøen en lengre og mer uttømmende liste. Krav slutter aldri å endre seg. Produktkøen er et levende dokument. Endringer i forretningens krav, forhold i markedet, teknologi og bemanning forårsaker endringer i produktkøen. For å minimere mengden av unyttig arbeid detaljeres bare de høyest prioriterte kravene. De kravene som teamet vil befatte seg med i de kommende sprintene er finkornede etter å ha blitt brutt ned på en slik måte at alle kan ferdigstilles i løpet av en sprint.

#### Tips

Oppgaver i produkt-backloggen er vanligvis utformet som User Stories. Use Cases er også hensiktsmessig, men egner seg kanskje enda bedre i utvikling av life- or mission-critical programvare.

#### Tips

Scrum-team bruker ofte så mye som 10% av hver sprint til jobbe med produkt-backloggen for å møte kravene definert i avsnittet over. Når den er brutt ned til denne finkornede tilstanden er oppgavne øverst i produktbackloggen (altså de med høyest prioritet, størst verdi) egnet til å passe inn i en sprint. De er blitt analysert og diskutert i denne fasen, og når sprint-planleggingen gjennomføres, er det ingen som ikke forstår disse høyt prioriterte oppgavne i produkt-backloggen og hvorfor de velges ut.

Flere Scrum-team jobber ofte sammen på det samme produktet. Én produktkø brukes til å beskrive det kommende arbeidet på produktet. En attributt ved produktkøen som grupperer elementer blir så brukt. Gruppering kan skje på bakgrunn av funksjonalitet, teknologi eller arkitektur og brukes gjerne som en måte for å organisere arbeidet til et Scrum-team.

Release-burndown er et diagram som gjenspeiler summen av gjenværende krav fra produktkøen estimert over tid. Estimater er i den enheten Scrum-teamet og organisasjonen har bestemt seg for. Denne enheten er vanligvis sprinter.

Estimater for krav i produktkøen beregnes først under release-planleggingen, og så etterhvert som kravene oppstår. Ved strukturerte gjennomganger av produktkøen blir de revidert, men de kan også oppdateres på et hvilket som helst tidspunkt. Teamet er ansvarlig for alle estimater. Produkteieren kan påvirke teamet ved å bedre forståelsen og ta avveininger, men det endelige estimatet blir utformet av teamet. Produkteieren har en oppdatert visning av produktkøen med release-burndown tilgjengelig til enhver tid. En trendlinje kan tegnes opp basert på endringer i det gjenværende arbeidet.

#### Tips

Akseptanse-tester blir også brukt til å jobbe med produkt-backloggen. De kan supplere den detaljerte beskrivelsen med en testbar funksjon av hva som denne oppgaven i produkt-backloggen er ment å løse.

#### Tips

I noen organisasjoner blir det lagt til fler oppgaver i backloggen enn det som ferdigstilles. Dette gjør at trendlinjen forblir flat eller til og med stiger. For å kompensere for dette og for å opprettholde gjennomsiktighet, defineres det et nytt gulv når oppgaver legges til eller tas ut. Det er kun svært viktige oppgaver som får definere et nytt gulv, og det må alltid dokumenteres grundig.

## SPRINTKØ OG SPRINT-BURNDOWN

Sprintkøen består av oppgavene teamet løser for å endre produktkøens krav om til et ferdig inkrement. Mange oppgaver er identifisert gjennom sprintplanleggingsmøter. Sprintkøen består av alle oppgaver som teamet identifiserer som nødvendige for å nå Sprint-målet. Oppgaver i sprintkøen må brytes ned til mindre oppgaver. Dette vil gjøre at endringer i prosessen kan måles og forstås på det daglige Scrum-møtet.

Teamet modifierer sprintkøen gjennom hele Sprinten, i tillegg til å behandle oppgaver som oppstår underveis i Sprinten. Etterhvert som teamet jobber med individuelle oppgaver, kan det innse at flere eller færre

oppgaver er nødvendig, eller at en bestemt oppgave vil ta mer eller mindre tid enn man har forutsatt. Etterhvert som nye oppgaver er påkrevet, legger teamet dette til i sprintkøen. Når oppgaver er under arbeid eller er ferdige oppdateres estimeringen av antall timer gjenstående arbeid. Når oppgaver blir vurdert som unødvendige fjernes de fra køen. Kun teamet kan endre på sprintkøen i løpet av en sprint. Kun teamet kan endre på innholdet eller estimatene. Sprintkøen er et svært synlig sanntidsbilde av det arbeidet teamet planlegger å utføre i løpet av sprinten, og eies i sin helhet av teamet.

Sprint-burndown er et diagram som viser mengden av arbeid som gjenstår i en sprint over tid. For å lage dette diagrammet bestemmes det hvor mye arbeid som gjenstår gjennom å summere opp estimatene i køen for hver dag i Sprinten. Mengden av arbeid som gjenstår i en sprint er summen av arbeid som gjenstår i hele sprintkøen. Hold oversikt over disse summene per dag og bruk de til å

### Tips

Trendlinjen er upålitelig til å begynne med gjennom de to, tre første sprintene med release, med mindre teamet ikke har jobbet med hverandre tidligere, er godt kjente med produktet og har dyp innsikt i teknologien.

### Tips

Prøv til enhver tid å vise en håndtegnet burndown-graf på et stort stykke papir og som er godt synlig i lokalet hvor teamet jobber. Det er mer sannsynlig at teamet legger merke til denne grafen enn det de gjør ved å kikke på et Excel-ark utledet av Sprint Burndown eller et annet verktøy.

lage et diagram som viser gjenstående arbeid over tid. Ved å dra en linje mellom punktene på diagrammet kan teamet overvåke sin fremdrift for gjennomføringen av oppgavene i sprinten. Varighet blir ikke vurdert i Scrum. Gjenstående arbeid og datoer er de eneste variabler av interesse.

En av Scrums regler som gjelder for hver Sprint er å levere inkremitter av potensielt leverbar funksjonalitet som oppfyller definisjonen av "ferdig".

## FERDIG

Scrum fordrer at teamet bygger et inkrement av ferdig funksjonalitet i hver sprint. Dette inkrementet må være leveringsklart, for produkteieren kan velge å produksjonssette funksjonaliteten med en gang. For å kunne gjøre dette må inkrementet være som en komplett del av det hele, det må være "ferdig". Hvert inkrement bør være forbedrende til alle tidligere inkremitter og må være nøye testet for å forsikre at alle inkremitter fungerer sammen.

### Tips

Uferdig arbeid blir ofte akkumulert i Product Backloggen som uferdig arbeid eller "Implementation Work". Når disse oppgavene samles og akkumuleres fører det til at Product Backloggen blir mer nøyaktig og gjengir virkeligheten.

I produktutvikling kan det å hevde at funksjonaliteten er ferdig lede noen til å anta at produktet er godt kodet, forbedret, enhetstestet, bygget og akseptansetestet. Atter andre kan anta at det kun er koden som er bygget. Hvis ikke alle kjenner definisjonen av "ferdig" virker ikke de andre to pilarene av den empiriske prosesskontrollen. Når noen beskriver noe som "ferdig" må alle ha en felles oppfatning av hva *ferdig* betyr.

"Ferdig" definerer hva teamet mener når de påtar seg å implementere et krav fra produktkøen i en sprint. Noen produkter har ikke dokumentasjon, så i definisjonen av "ferdig" inngår ikke dokumentasjon. Et fullstendig "ferdig" inkrement inkluderer all analyse, design, forbedring, programmering, dokumentasjon og testing for dette inkrementet og for alle elementer i produktkøen for inkrementet. Testing inkluderer enhets-, system-, bruker- og regresjonstesting, i tillegg til ikke-funksjonelle tester som ytelses-, stabilitets-, sikkerhets- og integrasjonstesting. "Ferdig" innbefatter også eventuell internasjonalisering. Noen team kan enda ikke inkludere alle momenter nødvendig for produksjonssetting i deres definisjon av ferdig. Dette må gjøres klart ovenfor produkteieren. Det

gjenstående arbeidet må utføres før produktet kan produksjonssettes og brukes.

## OPPSUMMERING

Noen organisasjoner er ikke i stand til å bygge et komplett inkrement i løpet av en sprint. De har kanskje ikke infrastrukturen for automatisert testing som er nødvendig for å gjøre ferdig all testing. I et slikt tilfelle lages det to kategorier for hvert inkrement; det "ferdige" arbeidet og det "uferdige" arbeidet. Det "uferdige" arbeidet er den delen av hvert inkrement som vil måtte ferdigstilles på et senere tidspunkt.

Produkteieren vet nøyaktig hva hun eller han skal inspisere på slutten av sprinten fordi inkrementet oppfyller statusen "ferdig" og produkteieren forstår definisjonen. "Uferdig" arbeid legges til i en produktkø kalt "Uferdig arbeid", slik at det akkumulerer seg der og blir korrekt reflektert i release-burndown. Denne teknikken gir gjennomsiktighet i prosessen mot en release. Delen av en sprintgjennomgang hvor man inspiserer og tilpasser blir like nøyaktig som denne gjennomsiktigheten.

Hvis et team for eksempel ikke er i stand til å utføre ytelses-, regresjons-, stabilitets-, sikkerhets- og integrasjonstesting for hvert krav i produktkøen vil andelen av dette arbeidet i forhold til arbeidet som kan utføres (analyse, design, forbedring, programmering, dokumentasjon, enhets- og brukertesting) bli kalkulert.

La oss si at denne andelen består av seks deler "ferdig" og fire deler "uferdig". Hvis teamet ferdigstiller et krav i produktkøen som består av seks oppgavedeler (teamet estimerer på basis av hva det kan utføre), blir fire deler lagt til i "uferdig arbeid" i produktkøen. Sprint for sprint akkumuleres de "uferdige" oppgavene i hvert inkrement. Disse må tas hensyn til i forkant av release for produktet. Dette arbeidet akkumuleres lineært selv om det faktisk har en slags eksponensiell akkumulering som er avhengig av hver organisasjons egenart. En egen release-sprints legges til enden av enhver release for å ferdigstille "uferdige" oppgaver. Antall sprinter for dette er like uforutsigbart som akkumuleringen av "uferdig arbeid" ikke er lineær.