AARHUS
UNIVERSITET
INGENIØRHØJSKOLEN

Vejledning til udviklingsprocessen

for semesterprojekt 3 (PRJ3)

Version 2.00 / 14-09-2016 / PHM

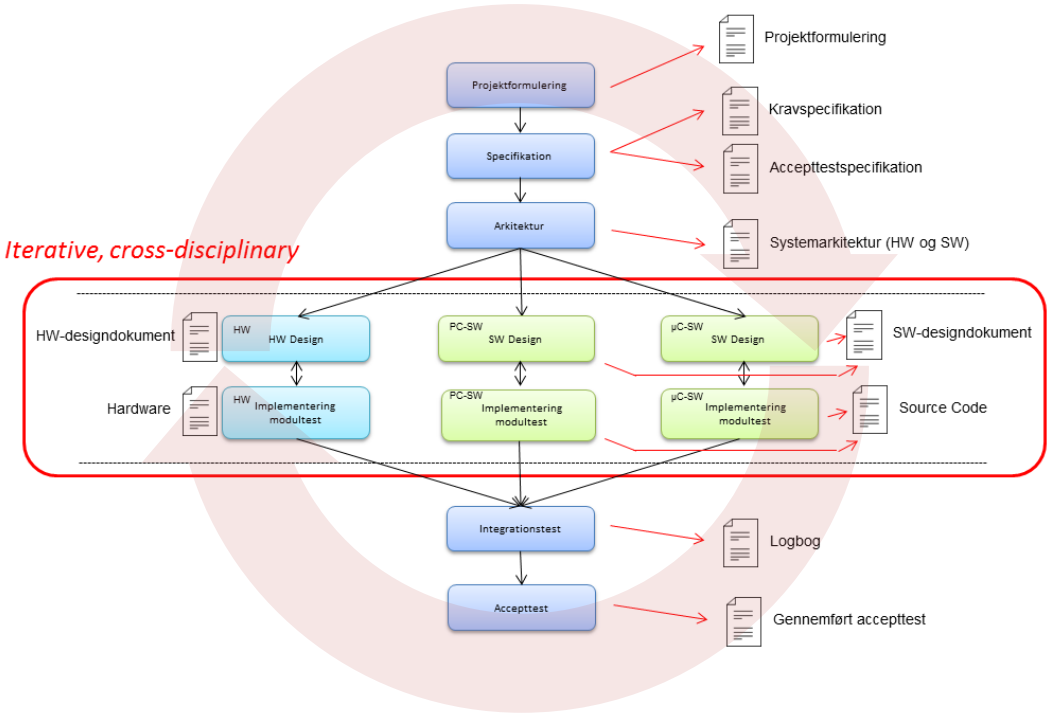
Indholdsfortegnelse

Indledning

Denne vejledning har til formål at beskrive udviklingsprocessen, som følges på 3. semesters semesterprojekt. Projektet er fælles for E-, EP- og IKT-studerende på diplomingeniøruddannelsen på Aarhus Universitet. Fokusområdet er udvikling og implementering af et selvvalgt produkt, med udgangspunkt i en given hardwareplatform. Udviklingsprocessen gennemføres ved hjælp af iterative metoder og på baggrund af de procesmetoder, de studerende har lært på 2. semester. Projektets indhold er detaljeret i "Projektoplæg for 3. semester"[1].

Baggrund

"ASE-udviklingsmodellen"[2] vist i Figur 1, er introduceret på 2. semester og giver en struktureret model for udvikling af et system bestående af hardware og software.



Figur 1 ASE-Modellen (Kilde: Kim Bjerge, Vejledning til udviklingsprocessen for projekt 2)

Modellen er velegnet til udvikling af soft- og hardware for projekter hvor man har et godt kendskab til domæne, teknologi, implementeringsmuligheder mm. Med et godt kendskab til disse er projektrisici lave og det er muligt at beskrive præcise krav, formulere tests og designe komplette use-cases på et tidligt tidspunkt. Ofte byder projekter dog på projektrisici af forskellig art, men ved at kombinere ASE-modellen med en iterativ proces, kan vi gøre modellen mere robust overfor disse og fremtidige tilpasninger.

Iterativ udvikling er kendetegnet ved at man gennemfører sit projekt i små bidder (iterationer), som hver især resulterer i et del-produkt som virker. Derigennem opnår man tidligt tillid til at projektet kan bringes til at virke, samt lavet en tidlig risikoafdækning.

Hver iteration indeholder elementer fra flere af processerne beskrevet i ASE modellen. En iteration kan således indeholde både design, udvikling og test, men kun for en mindre del af projektet.

[1] Ingeniørhøjskolen, Aarhus Universitet, "Projektoplæg for semesterprojekt 3", August 2015
[2] Ingeniørhøjskolen, Aarhus Universitet, "Vejledning for gennemførelse af projekt 2", August 2015

Projektet på 3. semester indeholder mange nye teknologier og dermed mange tekniske risici, samt de projektmæssige, som følge af projektarbejdet, og det er derfor en god idé at bruge en iterativ arbejdsmetode.

Iterativ udvikling med ASE-modellen

Selvom man vælger en iterativ proces til udviklingen af et projekt, har projektet stadigvæk forskellige faser, som vist på figur 2. Fokus for de første iterationer er primært ideudvikling og fastlæggelse af krav. I de efterfølgende iterationer flytter fokus sig til regulært udviklingsarbejde og senere til produktoverlevering. Forskellen på en iterativ udviklingsproces og ASE-modellen i Figur 1, er at vi arbejder parallelt på flere typer opgaver indenfor hver iteration. Vi gør dette for løbende at kunne bygge, ændre og forbedre produktet på baggrund af erfaringer og erkendelser fra tidligere iterationer.

Vi indleder for eksempel projektet med at opstille de første og vigtigste krav for projektet. Når disse krav er lagt fast, begynder vi at grave os ned i problemdomænet, f.eks.: ”Hvordan måles en afstand uden berøring?”, ”Hvordan implementeres en GUI på en embedded platform?” Dette arbejde kvalificerer de opstillede krav, lader os uddybe projektformuleringen og giver mulighed for at lave fysiske tests.

I hver iteration arbejdes på næsten alle projektdiscipliner. Dokumentation udvikler sig i løbet af projektforløbet og skal derved opdateres løbende.

Udviklingsprocessen i PRJ3

Projektet på 3. semester udvikles med de processer og artefakter, som også blev anvendt i projektet på 2. semester, mens Scrum anvendes til at styre den iterative arbejdsgang. Fordelen ved at anvende en kendt arbejdsmetode som Scrum, er at arbejdsgange og roller er velbeskrevne. Den strukturerede arbejdsgang vil hjælpe med at gøre projektet mere overskueligt, hjælpe til at holde projektet i gang, selvom andre fag presser på og ikke mindst gøre det sjovere, da der kan arbejdes mod små tydelige mål. Desuden er Scrum udbredt i industrien, så de studerende opnår også erhvervsrelevant erfaring med anvendelsen af Scrum.

Projektet definerer en ramme med et sæt milepæle med obligatoriske delafleveringer, men gruppen bestemmer selv, hvordan den vil møde disse milepæle. I de følgende afsnit er der givet et forslag til hvordan gruppen, især i opstarten af projektet, kan strukturere sin arbejdsproces. Opstarten af et udviklingsprojekt er altid vanskelig og kræver styring og overblik, for at undgå at spilde meget tid af på det forkerte.

Milestones

Der er følgende milestones for PRJ3 (relativt til semester start):

- Uge 2: Gruppedannelse afsluttes
- Uge 3: Projektformulering afleveres til vejleder
- Uge 7: Review af kravspecifikation, testspecifikation og systemarkitektur (Inden påbegyndelse af konstruktions Sprints)
- Uge 13: Aflevering af projektartefakter

Iterationer

Der foreslås i det følgende iterationer til gennemførsel af projektet. Omkring opstart handler opgaverne primært om at få et overblik over projektet. Senere går man mere over til produktion af kode og hardware. I bilagene findes oplæg til Task Boards, som kan anvendes i Scrum.

Iteration #1: Formulering af projekt (1 uge)

I den første fase er målet at opnå klarhed om projektidé, de vigtigste krav, samt væsentlige tekniske risici.

- **Projektformulering**: Projektformuleringen formuleres som kendt fra vejledningen for 2. semester og gerne assisteret af "rige" billeder. Der formuleres et mål for projektet som alle krav efterfølgende støtter op om, f.eks: "En ske, som gør det muligt for parkinson-patienter at spise selv". Der inkluderes også en MosCoW analyse til at identifikation og prioritering af systemkrav.
  Kravspecifikation: Nogle få essentielle "Must-have" krav udvælges og beskrives som uformelle ("brief") use cases og tilhørende ikke-funktionelle krav. Disse krav udvælges på baggrund af, hvor kritiske de vurderes at være for systemets samlede funktionalitet.
  Systemarkitektur dokument: Beskriv en domænemodel på baggrund af de Use Cases som er valgt til første iteration. Lav en overordnet teknisk risikoanalyse på systemniveau. Denne skal identificere tekniske risici, som er kritiske for projektet. F.eks. ukendte interfaces, ukendte programmeringsframeworks, ukendte teknologier, ting som potentielt ikke vil spille sammen etc.
  Projektplan: Udled en grov overordnet skitse med angivelse af milestones og iterationer

Iteration #2: Afdækning og minimering af risici (2 uger)

Denne næste fase har som mål at identificere og om muligt minimere risici, ved at gøre projektgruppen bekendt med problemdomænet samt de teknologier og metoder som der skal arbejdes med.

Det er meningen, i denne fase, at projektgruppen gør sig konkrete erfaringer med problemdomæne, arkitekturer, kode, hardware, algoritmer mm., for at afdække risici og tilegne sig viden om ting der er ukendte, og som dermed giver de største tekniske risici. Dette tillader en mere nøjagtig estimering af opgavers omfang i fremtidige iterationer.

Til forskel fra 2. semester forventes det ikke at gruppen har færdig kravspecifikation før arbejdet med fastlæggelsen af en systemarkitektur påbegyndes. 3. semesterprojektets mere udforskende karakter gør det nødvendigt tidligt at fastlægge det bedste bud på en arkitektur og herefter løbende gøre sig erfaringer, som bl.a. vil betyde en løbende revision og detaljering af kravspecifikationen.

- **Systemarkitektur**: Fasen indledes med en analyse af hvad der kræves af HW, SW og mekanik for at reducere risici. Fasen afsluttes med skitsering af systemets overordnede arkitektur, herunder blokdiagrammer (BDDs) for logiske blokke, overordnede interne blokdiagrammer (IBDs) og identifikation af interne interfaces med sekvensdiagrammer. Endvidere påbegyndes en applikationsmodel for hver CPU i systemet med udgangspunkt i de Use Cases som er taget med i først omgang. Applikationsmodellen skal i denne iteration indeholde et klasse diagram for hver CPU med boundary, control og domain klasser.
  Implementering og Test: Design defineres og implementeres vha. mock-ups, evaluation boards, virtuelle maskiner, emulatorer etc. i den udstrækning det er nødvendigt for hurtigt at nå frem til et proof-of-concept-system som sandsynliggør at den valgte arkitektur kan anvendes og/eller drage erfaringer til ændringer af denne.
  Kravspecifikation: Fasen afsluttes med en opdatering og præcisering af krav. Få, men essentielle, use cases laves fully-dressed og ikke-funktionelle krav som eksterne interfaces, miljø mm præciseres.
  Projektplan: På baggrund af erfaringerne opdateres projektplanen.

Iteration #3: Tidlig implementering og stabilisering af krav (2 uger)

I denne fase flytter fokus sig fra at opnå domænekendskab til at designe og implementere løsninger til opfyldelse af projektets krav. Der arbejdes således ud fra use cases og ikke-funktionelle krav.

Udviklingen er på et meget tidligt stadie og meget kan og vil ændre sig i denne fase mht. krav, arkitektur og design. Udviklingen centrerer sig omkring ikke-funktionelle krav og Use Cases, eller dele af disse, som hjælper os med at teste om den foreslåede hardware- and software arkitektur, er hensigtsmæssig og lever op til vores krav. Man kan ikke forvente at afdække det 100 %, men man kan få en tidlig indikation.

Således er det umiddelbare produkt af denne fase ganske lille. Typisk er få procent af det færdige projekt bygget ved afslutning af denne fase. Fasens egentlige produkt er nemlig risikoafdækning, -adressering og -minimering, samt stabilisering af krav.

- **Kravspecifikation**: Fasen afsluttes med opdatering og præcisering af oprindelige og nye krav. De næste Use Cases kan nu gøres formelle. Kravene bør være relativt stabile nu.
- **Systemarkitektur:** Fasen indebærer en analyse af hvad der er nødvendigt for at implementere de krav som giver mest værdi. Denne analyse resulterer i en opdatering af systemarkitekturen, dokumenteret ved BDDs, IBDs, interne interface-beskrivelser og SW arkitektur. Applikationsmodellerne for hver CPU opdateres med sekvensdiagrammer, revideret klassediagrammer og evt. statmachines hvis I finder dem nødvendige. Efter denne fase bør de centrale elementer i systemarkitekturen være rimelig stabile.
- **HW/SW Designspecifikation:** Der udarbejdes et design for realiseringen af de udvalgte krav.
- **Implementering og Test:** Det udarbejdede design implementeres og modultestes.
- **Projektplan:** På baggrund af erfaringerne opdateres projektplanen.

Iteration #4-(N-1): Konstruktion af projekt (2 uger per sprint)

I denne fase er fokus på fremstilling af systemet. De primære opgaver er design, implementering og test af systemet til imødekommelse af de opstillede krav. Kravene til disse iterationer prioriteres og udvælges efter deres "business value", dvs. den værdi opfyldelsen af dem tilfører produktet.

Fasen opdeles i iterationer af 2 ugers varighed. Før en iteration påbegyndes, planlægges de opgaver som søges løst i iterationen. Hver opgave skal løses fuldkomment i løbet af sprintet, dvs. de systemkomponenter som skal skabes eller udvides for at løse opgaven, skal designes, implementeres og modultestes. En integration af disse komponenter i systemet bør også finde sted i den indeværende iteration.

Det er en stor fordel at definere et klart mål for iterationen, som demonstreres opfyldt ved afslutningen af en iteration. Det er langt sjovere og mere overskueligt at arbejde mod et minimalt, men fungerende GUI som kan styre en simpel LED, end at arbejde mod at implementere 10 % mere af det komplette (uoverskuelige) system.

- **Kravspecifikation**: Krav opdateres om nødvendigt. Rettelser til kravene vil typisk være af mindre, uddybende karakter.
- **Systemarkitektur:** Opdater ved ændringer i HW/SW arkitektur og interfaces. Arkitekturændringer og –tilføjelser forårsaget af de fremstillede systemkomponenter og deres integration med systemet dokumenteres løbende.
- **HW/SW Designspecifikation:** Design for de næste krav, medtag erfaringer fra sidste iteration. Designændringer og – tilføjelser forårsaget af de fremstillede systemkomponenter og deres integration med systemet dokumenteres løbende.
- **Implementering og Test:** De fremstillede systemkomponenter implementeres og testes løbende, både som isolerede moduler og som integrerede moduler i systemet.
- **Projektplan:** Der bør på dette tidspunkt ikke være ændringer til tidsplanen. Iterationerne i planen ligger fast – *indholdet* af disse er fleksibelt, men *længden* af dem ændres ikke.

Iteration #5: Overlevering af projekt (1-2 uger)

I projektets sidste fase skal implementering og projektdokumentation afsluttes, således at den kan overgives til tredjepart.

- **Kravspecifikation**: Opdateres og afsluttes.
- **Systemarkitektur:** Opdateres og afsluttes.
- **HW/SW Designspecifikation:** Opdateres og afsluttes.
- **Implementering og test:** Implementeringen og testen af systemet afrundes. Hvis der er dele af projektet som ikke er nået, og der som konsekvens heraf er dele af systemet som ikke er bygget, eller bygget delvist, stubbes disse dele af.
- **Projekt- og procesrapport:** Disse dokumenter færdiggøres.

Scrum i PRJ3

Der skal anvendes Scrum til opgavestyring i PRJ3. Det forventes at den studerende har et grundlæggende kendskab til Scrum og **der henvises generelt til den originale (obligatoriske) "Scrum Guide" (http://scrumguides.org/), den er ganske let læst!**

I Scrum er der veldefinerede artefakter, roller og opgaver, og enkelte af disse er i PRJ3 justeret til en studiemæssig kontekst. De følgende afsnit giver et kort overblik over disse, samt hvordan de fungerer i PRJ3.

Artefakter i Scrum

Scrum definerer en række artefakter, som anvendes i projektet:

- *Product Backlog* (PB) er et prioriteret "katalog" med alle potentielle opgaver ("items") i hele projektet. PB'en prioriteres og raffineres i forbindelse med Sprint Planning (s.d.).
- *Sprint Backlog* (SB) er opgaver til løsning i det indeværende sprint. Der findes kun en SB for det indeværende sprint, ikke for efterfølgende sprints. Opgaverne ("items") i SB'en udtages og estimeres fra product backloggen ifm. Sprint planning. De enkelte opgaver ("items") i SB'en er nedbrudt i mindre (typisk teknisk fokuserede) *tasks*, hver især af ½-1 dags varighed for 1 person.
- *Task Board'et* er et whiteboard eller lignende som til enhver tid afspejler status for det indeværende sprint. Task boardet viser Sprint Goal, Burndown Chart og opgaverne i Sprint Backlog'en samt status for disses individuelle tasks: New, In Progress eller Done.

Roller i Scrum

De fastlagte roller i Scrum giver en klar ansvarsfordeling i arbejdet og er beskrevet i det følgende.

Scrum Master

Scrum Masteren (SM) er ansvarlig for at Scrum anvendes korrekt af Development Team'et. SM tager initiativ til Scrum møder, faciliterer teamets arbejde, fjerner forhindringer for projektet etc. Bemærk: En SM er *ikke* projektleder eller sekretær!

I PRJ3 vælger gruppen selv sin SM. Rollen kan eventuelt gå på omgang efter sprints. SM deltager også som Team Member i Development Teamets udviklingsarbejde. Opgaver som mødereferat mm. Fordeles i gruppen. SM har typisk kontakten til vejleder.

Product Owner

Product Owner (PO) er aftagernes repræsentant. Han har som mål at opnå det bedst mulige produkt, og har typisk domænekendskab, men ikke nødvendigvis dyb teknisk forståelse. PO prioriterer, raffinerer og trimmer Product Backlog, således at målet med disse er tydelige for hele gruppen.

I PRJ3 udgøres PO af en virtuel person som varetages af projektgruppen, f. eks. en bamse på væggen, som forsvarer det oprindelige overordnede projektmål (f.eks. "En ske, som gør det muligt for parkinson-patienter at spise selv"). Hvis projektgruppen er i tvivl om mening med-, eller prioritering af opgaver, skal de således kunne sætte sig i PO's sted og handle som han/hun ville have gjort.

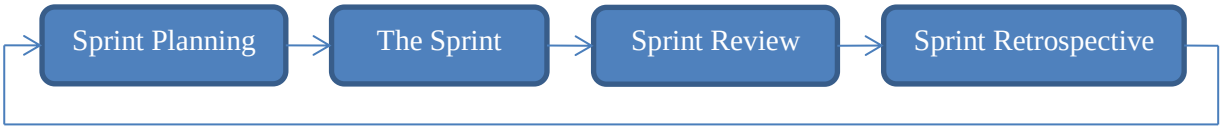
Development Team, Team Member

Arbejdet med at realisere backlog items udføres af Development Team'et (DT'et). Medlemmerne af DT'et kaldes Development Team Members (DT members). DT members er ansvarlige for at afklare, konkretisere og estimere opgaverne i Product Backlog'en ifm. Sprint Planning, og for at realisere (dvs. designe, implementere og teste løsningen til) opgaverne

I PRJ3 er alle projektgruppens medlemmer DT Members. SM har som nævnt yderligere opgaver.

Aktiviteter i Scrum

Der er veldefinerede aktiviteter i Scrum, der hver især har specifikke formål: Nedbryde opgaver, løse dem og lave opsamling som vist i Figur 2. Disse opgaver løses i hvert Sprint



Figur 2 Sprint Cycle

Sprint Planning

Målet for denne aktivitet er at få afklaret og estimeret opgaver fra Product Backlog'en til løsning i det næste sprint, at fastlægge hvordan arbejdet skal udføres (dvs. f. eks. overordnede arkitektur-beslutninger), og at fastlægge et Sprint Goal.

PO foreslår et mål for det kommende sprint og foreslår opgaver fra Product Backlog, som understøtter dette. Hele teamet samarbejder om at nedbryde opgaver og estimere deres omfang. Opgaverne fra PB'en lægges nu i det kommende sprints Sprint Backlog, så det samlede omfang af opgaverne netop fylder lige så meget, som man nåede i det forgangne sprint. Opgaverne nedbrydes i tasks af cirka ½-1 arbejdsdags varighed. Aktiviteten afsluttes med at gruppen samlet definerer et klart Sprint mål.

I PRJ3 varetager projektgruppen PO's rolle (Se afsnit om roller). En kopi af det planlagte Sprint Board inklusiv et tydeligt Sprint Goal sendes til gruppens vejleder for kommentarer. Projektgruppen behøver ikke vente på vejleders kommentar før arbejdet påbegyndes. Gruppen kan forvente svar fra vejleder indenfor få dage. Sprint Goal er det som vejleder forventer at få demonstreret til den efterfølgende Sprint afslutning.

The Sprint

I denne aktivitet udføres selv udviklingsarbejdet. Målet med sprintet er at producere "a potentially shippable" delprodukt, som defineret af sprint goal'et. Et sprints varighed er typisk 2 uger. I løbet af sprintet tager udviklere konstant de højest prioriterede tasks fra Sprint Backlog'en og designer, implementerer

og tester disse. En Task overdrages typisk *ikke* til et andet DT Member under udførelsen af det – udvikler er altså selv ansvarlig for gennemførelsen af opgaven, men kan selvfølgelig søge hjælp og assistance. I udførelsen af en Task har kvalitet højest prioritet, hvorfor det prioriteres at afslutte en opgave fuldstændigt, fremfor at have to opgaver hængende ved en sprint afslutning. Hvis man gør sig erfaringer, som kræver ændringer i opgaver mens sprintet kører, aftales dette med PO og afspejles i Sprint Backlog'en.

I PRJ3 kører projektgruppen selv denne aktivitet efter bogen[3]

Daily Scrum

Målet med denne aktivitet er at holde teamet og SM opdateret med opgaver, forhindringer m.m. i det igangværende Sprint. Mødet er et dagligt 15-minutters møde med agendaen:

- Hvad lavede du i går, som hjælper til at nå Sprint målet?
- Hvad skal du lave i dag, som hjælper til at nå Sprint målet?
- Ser du forhindringer i dit arbejde mod Sprint målet?

Den korte varighed er afgørende for mødets effektivitet.

I PRJ3 kan det være svært at finde tid og sted til at mødes dagligt. Det anbefales at holde et kort møde svarende til Daily Scrum ca. 2 gange om ugen, ud over den tid man sidder sammen og arbejder. Mødet kan eventuelt foregå via Skype eller på en Facebook chat. Mødet er dog vigtigt, for at mindske misforståelser og for at have en fælles forståelse af hvad der arbejdes på.

Sprint Review

Dette møde afholdes ved afslutningen af et Sprint, og alle er velkomne til dette møde. Målet med mødet er at demonstrere *produktet* og derigennem afgøre hvilke opgaver der er afsluttede til PO's tilfredshed. Desuden skal Product Backlog opdateres, således den står klar til den efterfølgende Sprint Planning. I løbet af mødet skal:

- Teamet demonstrere hvad de har lavet
- PO afgør hvilke opgaver der er afsluttede
- Tekniske udfordringer i forbindelse med det gennemførte sprint, samt løsninger til disse, diskuteres

I PRJ3 gennemfører gruppen selv dette møde. Vejleder kan deltage efter ønske fra gruppen.

Sprint Retrospective

I dette møde ses tilbage på *processen* (ikke produktet) i det forgangne Sprint og det identificeres hvad som fungerede godt i processen og hvad der skal gøres anderledes i det kommende Sprint. Mødet er også en god anledning til at se på det foregående sprints Burn Rate, dvs. hvordan gik det reelt med at få løst opgaverne i det forgangne sprint og hvor mange Story Points blev dermed indløst. Dette tal kaldes teamets velocity og anvendes til at afgøre, hvor mange opgaver teamet kan påtage sig i det kommende sprint (se "Sprint Planning").

Mødet afholdes med følgende agenda:

- Hvad skal vi som team …
  - blive ved med at gøre (Eks: Fast tidspunkt for stå-op møde, 2-ugers sprints)

[3] "Scrum Guide" (http://scrumguides.org/)

  - holde op med at gøre (Eks: Aflevere ikke-testede opgaver…)
  - Begynde at gøre (Eks: Intern demo af features før kunden ser den)
- Hvad har vi som team gentagne problemer med?(Eks: Vi påtager os uafklarede ekstraopgaver i løbet af sprintet)

Hvis der identificeres action items, der er tidskrævende at implementere, formuleres disse som Backlog Items og medtages i det kommende sprints Sprint Backlog.

I PRJ3 afholdes mødet sammen med vejleder. Vejleder kan eventuelt være mødeleder for at hjælpe gruppen med at udlede og udrede problemstillinger i processen.

Opsummering af aktiviteter og roller

I Figur 3 er vist en opsummering af de forskellige roller og aktiviteter i Scrum, som det bliver afviklet i PRJ3

Scrum Aktivitet	Team Member	Product Owner	Scrum Master	Vejleder
Sprint Planning	Udleder opgaver og vurderer tidsforbrug (Planning Poker)	Bestemmer over hvad som skal på task board	Støtter P.O. i at forfølge projektets mål og ellers som Team Member	Godkender tilsendt Taskboard og Sprint Goal
The Sprint	Design, implentering og test af opgaver som man selv har taget på Task Boardet		Som Team Member, men støtter teamet i at bruge Scrum og har kontakt til Vejleder	
Sprint Review	Afgøre hvad som er færdigt		Er mødeleder	Deltager i demonstration
Sprint Retrospective	Diskuterer proces		Diskuterer proces	Vejleder faciliterer mødet
Daily Scrum	Deltager		Arrangerer og er mødeleder	

Figur 3 Oversigt over aktiviteter og roller

Projektgruppen afvikler således selv de fleste aktiviteter, men ved sprint afslutning inviteres vejleder til et møde, hvor der gives en demonstration af hvad der er opnået i Sprintet, samt der gennemføres til Sprint Retrospective. På mødet kan målene for det kommende Sprint også diskuteres med vejleder. Efter mødet står gruppen selv for Sprint Planning, men sender en kopi af/link til Task Board samt Sprint mål til vejleder, som kommenterer på disse.

Bilag: Eksempel Task Boards

#1: Formulering af projekt (1 uge)	New	In Progress	Done
Projektformulering	Lav en tekstuel formulering af projekt i projektformuleringen		
	Lav et "Rigt" billede til beskrivelse af systemets funktionalitet i projektformuleringen		
	Lav MoSoW analyse i projektformuleringen		
Kravspecifikation	Beskriv få, men vigtige funktionelle krav vha. uformelle Use Cases i kravspecifikationen		
	Beskriv essentielle ikke funktionelle krav i kravspecifikation		
Risikoanalyse	Lav teknisk risikoanalyse på systemniveau og dokumentér i Systemarkitektur dokumentet		
Planlægning	Lav en overordnet projektplan i projektformuleringen		

#2: Afdækning og minimering af risici (2 uger)	New	In Progress	Done
Identifikation og minimering af risici	Mindsk hardware risici gennem implementering og test af udvalgte hardware interfaces/funktionalitet vha mock-ups, eval boards mm.		
	Mindsk software risici ved at små implementeringer og test som anvender de ukendte software frameworks, sprog, IDE'er, OS'er som overvejes til projektet.		
Analyse af System arkitektur	Nedskriv bedste bud på en overordnet software arkitektur ud fra erfaringer gjort i dette sprint		
	Nedskriv bedste bud på en overordnet hardware arkitektur ud fra erfaringer gjort i dette sprint		
Præcisering af krav	Udvalgte Use Cases udspecificeres som fully-dressed ud fra erfaringer gjort i dette sprint		
	Præcisér ikke-funktionelle krav ud fra erfaringer gjort i dette sprint		
Opdatering af Projektplan	Opdater projektplan ud fra erfaringer gjort i dette sprint		
Få styr på projektstyringsværktøj	Indfør første sprints i Redmine, få oprettet et repo og en filstruktur på dette		

#3: Tidlig implementering og stabilisering af krav (2 uger)	New	In Progress	Done
Use Case som underbygger at overordnede softwarearkitektur er OK	Design, udvikl og test delfunktionalitet A		
Hardware som underbygger at overordnede koncept er OK	Design, udvikl og test delkredsløb K		
Analyse af System arkitektur	Opdatér software arkitektur ud fra erfaringer gjort i dette sprint		
	Opdatér hardware arkitektur ud fra erfaringer gjort i dette sprint		
Præcisering af krav	Nye udvalgte Use Cases udspecificeres som fully-dressed ud fra erfaringer gjort i dette sprint		
	Ikke funktionelle krav præciseres ud fra erfaringer gjort i dette sprint		

#4..(N-1): Produktion (2 uger per sprint)	New	In Progress	Done
Use Case som understøtter Sprintmålet	Design, implementér og test delfunktionalitet [B….]		
	Design, implementér og test delfunktionalitet [C….]		
	Evt. Integrationstest med resultat fra tidligere produktions sprint		
Hardware som understøtter Sprintmålet	Design, implementér og test delkredsløb L		
	Design, implementér og test delkredsløb M		
Følg Scrum	Hold Daily Scrum møder		
	Hold Sprint Review til sidst og identificer hvad som er lavet, læg ikke-færdige opgaver tilbage i Backlog		
	Invitér vejleder til demo + Retrospective,		
Opdater dokumentation	Opdater Systemarkitektur, kravspecifikation mm.		

#N: Overlevering af Projekt (1-2 uger)	New	In Progress	Done
Projektrapport	Opdateres og afsluttes		
	Endelig gennemlæsning og korrektur		
	Check referencer		
	Upload til Digital Eksamen		
Procesrapport	Beskriv resultater og erfaringer af de enkelte sprints		
	….og meget mere…		
Design, Implementering og Test	Afrund implementering. Stub ikke afsluttede dele af.		