AARHUS
UNIVERSITET
INGENIØRHØJSKOLEN

Vejledning til udviklingsprocessen

for semesterprojekt 3 (PRJ3)

Version 1.10 / 24-08-2015 / PHM

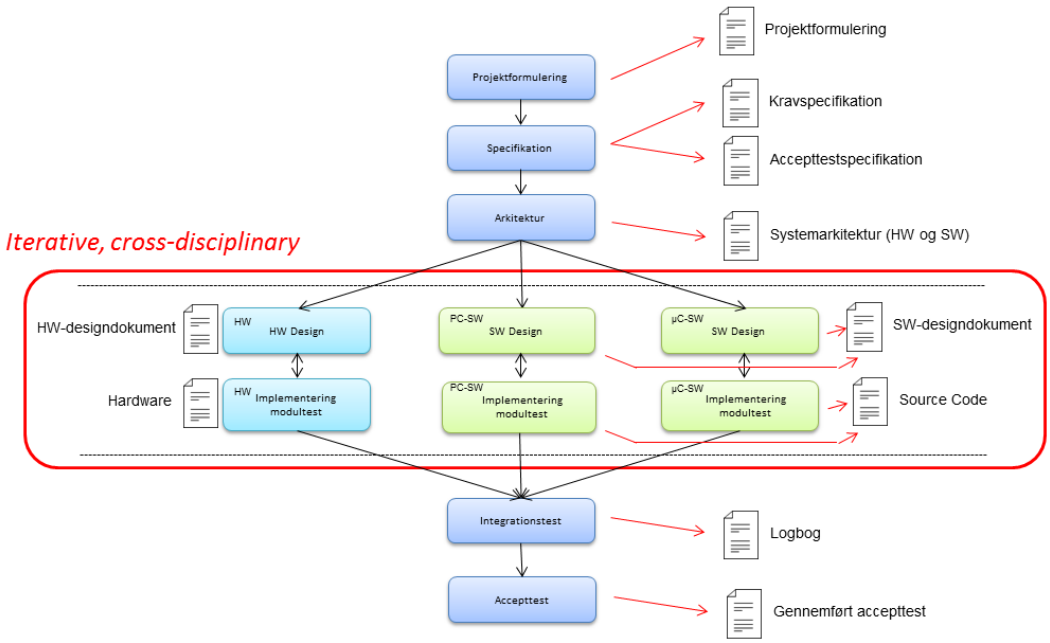
Indholdsfortegnelse

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 under anvendelse af procesmetoderne lært på studiets 2. semester. Projektets indhold er detaljeret i et Projektoplægget 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 ~~om~~ for hvilke man har et godt domænekendskab. Med et godt domænekendskab er det muligt at beskrive præcise krav, formulere tests og designe komplette use-cases på et tidligt tidspunkt. Ved at trække en iterativ arbejdsmetode ned over ASE-modellen, kan vi imidlertid gøre modellen mere robust overfor manglende domænekendskab og fremtidige tilpasninger.

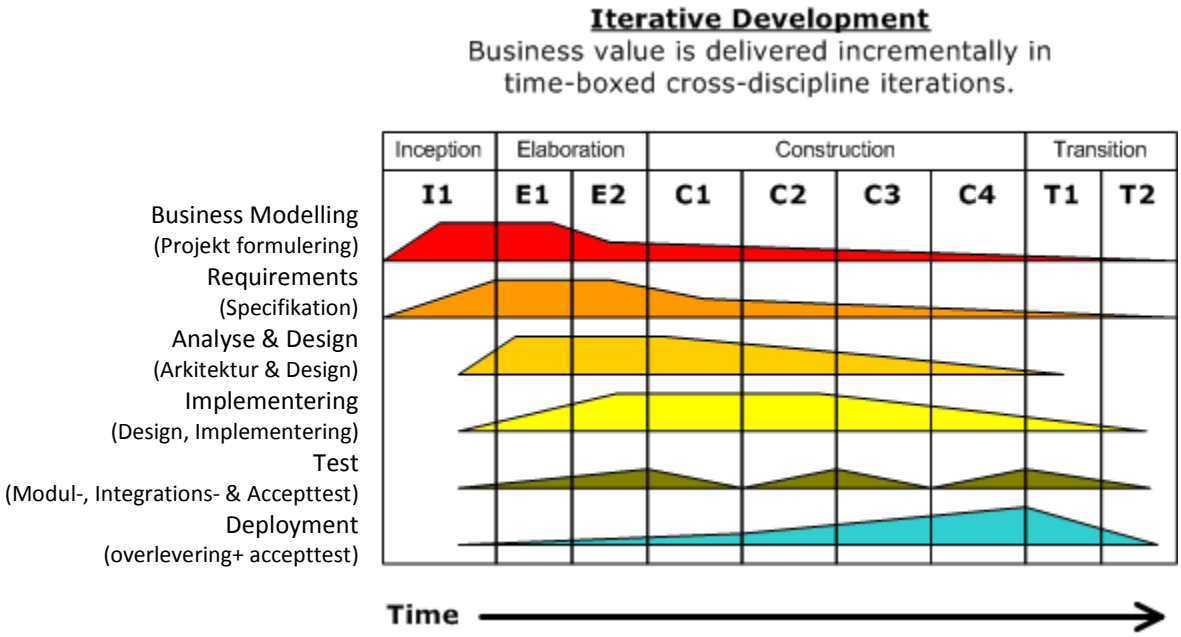
Iterativ udvikling er kendetegnende ved at man søger at gennemføre sit projekt i små bidder (iterationer). Hver iteration indeholder elementer fra flere af processerne beskrevet i ASE modellen. En iteration kan således indeholde både design, udvikling og test, men kun for en lille bid af projektet.

Projektet på 3. semester indeholder mange ubekendte og det er derfor en god idé at bruge en iterativ arbejdsmetode.

Iterativ udvikling med ASE-modellen

Selvom man vælger at arbejde iterativt, har projektet stadigvæk forskellige faser. Processerne omkring idéudvikling og kravspecifikation betones meget i de første faser, som vist i figur 2, mens detailudvikling fylder mindre. Betoningen forandrer sig gennem projektforløbet til at være mere udviklingsorienteret og senere orienteret mod produktoverlevering. Forskellen fra den oprindelige ASE-model i figur 1, er at vi arbejder parallelt på flere processor indenfor hver iteration. Vi gør dette for at bygge og forbedre 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 vi har styr på disse, begynder vi at grave os ned i problemdomænet, f.eks: ”Hvordan måles en afstand uden berøring?”, ”Hvordan implementeres et GUI på en embedded platform?” Processen kvalificerer kravspecifikationen, lader os uddybe projektformuleringen og giver mulighed for at lave fysiske tests.



Figur 2: Udviklingsfaser og Iterationer. ASE termer i parantes (Kilde: Wikipedia, Iterative and incremental development)

Et iterativt projektforløb kan indledes som vist i figur 3. (Inspireret af Larman: ”Applying UML and Patterns”, kap. 6). Bemærk hvorledes discipliner og artefakter mapper til de fra ASE-modellen i figur 1. Bemærk desuden at der arbejdes på næsten alle discipliner i hver iteration og at man allerede efter et par uger gerne forsøger sig med at ”prøve noget af” ved hjælp af mock-ups og lignende. Dokumentation udvikler sig i løbet af projektforløbet og skal derved opdateres løbende.

Iteration		1	2	3	4..n
Mål		Klarhed om projekt idé, samt de vigtigste krav	Opnå erfaring med problem-domænet for konkretisering af krav	Tidlig implementering og stabilisering af krav	Design, implementering og test af funktionalitet
Længde		1 uge	2 uger	2 uger	2 uger
Disciplin	**Artefakt**	**Inception 1**	**Elaboration 1**	**Elaboration 2**	**Construction 1..n**
Projekt Formulering	Projekt-formulering	Udled projekt-formulering. Anvend MosCoW til at identificere og prioritere krav.			
Specifikation	Kravspecifikation & Accept-test	Udvælg få "Must-have" krav der detaljeres som Use-Cases og ikke-funktionelle krav. Disse skal være dem som tilfører projektet mest værdi	Fasen afsluttes med en opdatering af krav. Halvdelen af Use-cases kan nu præciseres. Hardware blokke (BDD) og eksterne interfaces kan ligeledes præciseres.	Afsluttes igen med opdatering af krav og præcisering af yderligere Use-Cases. Ved Scrum er backlog ved at tage form nu.	Krav opdateres om nødvendigt, men disse burde være rimelig stabile nu. Ved Scrum reprioriteres backlog.
Arkitektur (System Analyse)	System-arkitektur	Ingen	Analysér hvad som skal til (hw/sw/mek) for at kunne implementere de få udvalgte krav	Analysér for de næste krav og opdater system arkitektur med BDD, eksterne interfaces og SW arkitektur	Opdater ved ændringer i sw arkitektur, eksterne interfaces.
Design	Design Dokumentation	Ingen	Lav design for de få udvalgte, men essentielle krav	Design for de næste krav	Design for de næste krav, medtag erfaringer fra sidste iteration
Implementering og modul test	HW/SW/ Mekanik	Ingen	Implementér disse. vha. mock-ups, eval boards, virtual machines, emulatorer etc. Således at vi får et proof-of-concept	Implementér og modultest disse. 5% bygget	Implementér og test disse. xx% bygget
Projekt Styring	Projekt plan	Udled grov overordnet plan	På baggrund af erfaringerne opdateres projektplan	Projektplan kan opdateres og er mere solid	

Figur 3: Eksempel på iterativ gennemførsel af projekt

Udvikling i PRJ3

Projektet på 3. semester udvikles med de processer og artefakter som også gjorde sig gældende i projektet på 2. semester.

Det iterative arbejde skal dokumenteres og det er vigtigt at man i starten af en iteration gør sig klart hvad målet for den pågældende iteration er, samt hvilke opgaver som skal søges løst af hvem. Ligeledes er det ved afslutning af en iteration vigtigt at gøre sig klart, hvad som er færdigt, dvs: Specificeret, designet, implementeret og testet, samt hvad som ikke er, inden man planlægger den næste iteration.

Det anbefales at anvende en struktureret metode til at implementere den iterative udvikling. Der findes mange forskellige metoder og det er op til projektgruppen at vælge en passende metode. Eksempler er Kanban og Scrum, som beskrives i afsnittet "Metoder til iterativ udvikling".

Milestones

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

- Uge 2: Gruppedannelse afsluttet
- Uge 3: Projektformulering afleveret til vejleder (efter *inception1*)
- Uge 7: Review af Kravspecifikation, testspecifikation og systemarkitektur (efter *elaboration2*)
- Uge 9: Review Implementation og proces (efter én *construction* iteration)
- Uge 14: Aflevering af projektartefakter

Udviklingsfaser

Udviklingsarbejdet tager udgangspunkt i fFigur 3 og anvender termerne herfra.

Inception

I denne første fase af projektet forventes det at projektgruppen når frem til en grundidé for projektet som ønskes gennemført.

Projektformuleringen skal formuleres som kendt fra vejledningen for 2. semester[2]. Til forskel fra 2. semester skal projektformuleringen dog også indeholde en MoSCoW analyse, som præciserer projektets mål; hvad som er essentielt og hvad som ikke er.

Projektformuleringen skal sammen med en første overordnet tidsplan godkendes af gruppens vejleder ved afslutning af iterationen.

Kravsspecifikationen påbegyndes som beskrevet i fFigur 3.

Elaboration

Denne næste fase har som mål at gøre projektgruppen bekendt med problemdomænet, samt at få en forståelse af hvilke teknologier og metoder som der skal arbejdes med. Fasen gør det desuden muligt bedre at vurdere kompleksitet og omfang – og dermed risici i projektet. Fasens iterationer tager udgangspunkt i centrale use-cases og ikke-funktionelle krav, hvis mulige løsninger undersøges ved hjælp af f.eks. mock-ups, tests på PC etc.

Til forskel fra 2. semester forventes det ikke at man har en komplet og færdig kravspecifikation før Systemarkitektur arbejdet påbegyndes. Som beskrevet i fFigur 3, evolverer kravspecifikationen over tid. I hver iteration udvælges de vigtigste ikke implementerede krav fra Projektformuleringen, som præciseres i kravspecifikationen som fully-dressed use-cases og ikke-funktionelle krav og som diagrammer, modeller og beskrivelser i systemarkitektur dokumentet.

Efter *Elaboration* fasen bør kravspecifikationen og systemarkitekturen være rimelig stabil.

Construction

I construction fasen er fokus at udvikle produktet. Fasen opdeles i iterationer af 2-3 ugers varighed. Før iterationen påbegyndes, planlægges hvilke opgaver, som søges løst. Hver opgave skal løses helt i løbet af sprintet, dvs. opgaven skal designes, implementeres og modultestes. Skal opgaven/delkomponenten integreres sammen med andre komponenter, bør dette også ske løbende, evt. som en ny opgave i den efterfølgende iteration.

Det er en stor fordel at definere et klart mål, som skal demonstreres ved afslutningen af en iteration. Det er langt sjovere og mere overskueligt f.eks. at arbejde mod et funktionelt, men minimalt, ~~gui~~GUI som kan styre en simpel LED, end at arbejde mod at implementere 10% mere af det totale (uoverskuelige) system.

Dokumentation

Det skal være muligt for vejleder og censor at vurdere gruppens iterative proces og gruppen skal derfor kunne dokumentere projektets status ved afslutning af hver iteration. Dette gøres nemmest ved at anvende et versionsstyringsværktøj til al elektronisk information. Herunder dokumenter, software, ~~hw~~ hardware-diagrammer, mødereferater etc. Repository skal være tilgængeligt for vejleder, således han ved gennemgang af versionshistorik kan danne sig et overblik over jeres proces. Det anbefales at der laves et tag/release for hver iteration, hvilket gør det betydelig lettere at overskue jeres proces.

Til forskel fra 2. semester (uddybes nedenfor):

- Skal al kode påføres forfatter
- Ingen funktionsdeklarationer i skriftelig kodedokumentationen
- Ingen skriftelig versionshistorik på dokumenter

Al kode skal have påført forfatter og hvis der anvendes kode udefra, skal dette angives med en kommentar i koden. F.eks: ”Tilpasset ud fra www.dfsdf.dk/post12”. Koden dokumenteres desuden vha. Doxygen tags, mens funktionsdeklarationer IKKE medtages i den skriftelige projektdokumentation.

Dokumenter kan skrives i Word, Latex mm efter eget valg. Latex giver dog visse fordele i forhold til versionsstyring, da formatet er ren tekst. Versionshistorik påføres ikke på forsiden af dokumenterne, i stedet påføres kun et versionsnummer på dokumentets forside og header/footer. Versionshistorikken påføres i stedet elektronisk i versionsstyringsværktøjet.

Metoder til iterativ udvikling

Der findes mange forskellige metoder til at udvikle iterativt. Metoderne beskriver arbejdsgange og typisk ikke et helt procesforløb. Nogle af de mest populære metoder pt. er Kanban og Scrum, som er beskrevet i det følgende.

Kanban

Er den oprindelige Toyota udviklingsmodel. Den har de seneste år fået en renæssance indenfor softwareudvikling, da det er en meget let model, som er nem at komme i gang med. Gode referencer er:

- Wikipedia: Kanban(development) – Fin introduktion. Se også henvisninger i bunden af siden (https://en.wikipedia.org/wiki/Kanban_%28development%29)
- OpenKanban - Introvideoer til Kanban udvikling og adgang til open source værktøj (http://agilelion.com/agile-kanban-cafe/open-kanban)

Scrum

Metoden udviklet til softwareudvikling i 1990'erne af Ken Swaber og Jeff Sutherland. Gode referencer er:

- The Scrum Guide - Guide af de oprindelige forfattere, nemt læst og overskuelig (http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf)
- Mountaingoat Software – Har et fint lille overblik over Scrum på deres hjemmeside (http://www.mountaingoatsoftware.com/agile/scrum/overview)
- Google Tech Talk: Scrum et al - Video med Kenn Swaber som fortæller om Scrum (https://www.youtube.com/watch?v=IyNPeTn8fpo)

Eksempel på brug af Scrum i PRJ3

PRJ3 er et typisk In-House udviklingsprojekt, hvilket vil sige at det er firmaet selv som definerer produktets mål. *Product Owner* rollen varetages derfor ikke af en ekstern kunde, men af teamet selv sammen med vejleder.

Den daglige rolle som *Scrum Master* bør varetages af et team medlem, som er i tæt kontakt med udviklingsteamet, hvilket vil sige et medlem af projektgruppen. Rollen kan evt. gå på skift mellem projektdeltagere.

Vejledningsmøderne afholdes typisk i forbindelse med sprint afslutning og indeholder følgende elementer:

- *Demonstation*, hvor gruppen demonstrerer hvad som er opnået i det forgangne sprint.
- *Sprint Retrospective,* hvor vejleder og gruppe ser tilbage på forløbet af det forgangne Sprint og identificerer hvordan det kan gøres bedre i den næste. Det være sig teknisk, procesmæssigt og interpersonelt.
- *Sprint Planning*, hvor vejleder og gruppe prioriterer tasks til den kommende Sprint samt evt. tilføjer nye tasks. Her skal sættes et klart og håndgribeligt mål for det kommende sprint. Eks: Detektér input og tænd LED.

Forud for mødet skal projektdeltagerne have afholdt et *Sprint Review* og opdateret *Product Backlog,* således at gruppen er klar til at planlægge næste sprint med vejleder. Opgaver som ikke er afsluttede i nærværende sprint, flyttes tilbage til *Product Backlog*'en.

Det anbefales at afholde flere ugentlige stå-op Scrum møder, f.eks. i pauserne eller evt. via Skype. Formålet med møderne er at ALLE ved hvad som sker på projektet og at opfange misforståelser, f.eks. omkring interfaces (HW/SW), deadlines og opgaver med mere.

Referencer

[1] Ingeniørhøjskolen, Aarhus Universitet, "Projektoplæg for semesterprojekt 3", August 2015

[2] Ingeniørhøjskolen, Aarhus Universitet, "Vejledning for gennemførelse af projekt 2", August 2015