# Projekt beskrivelse

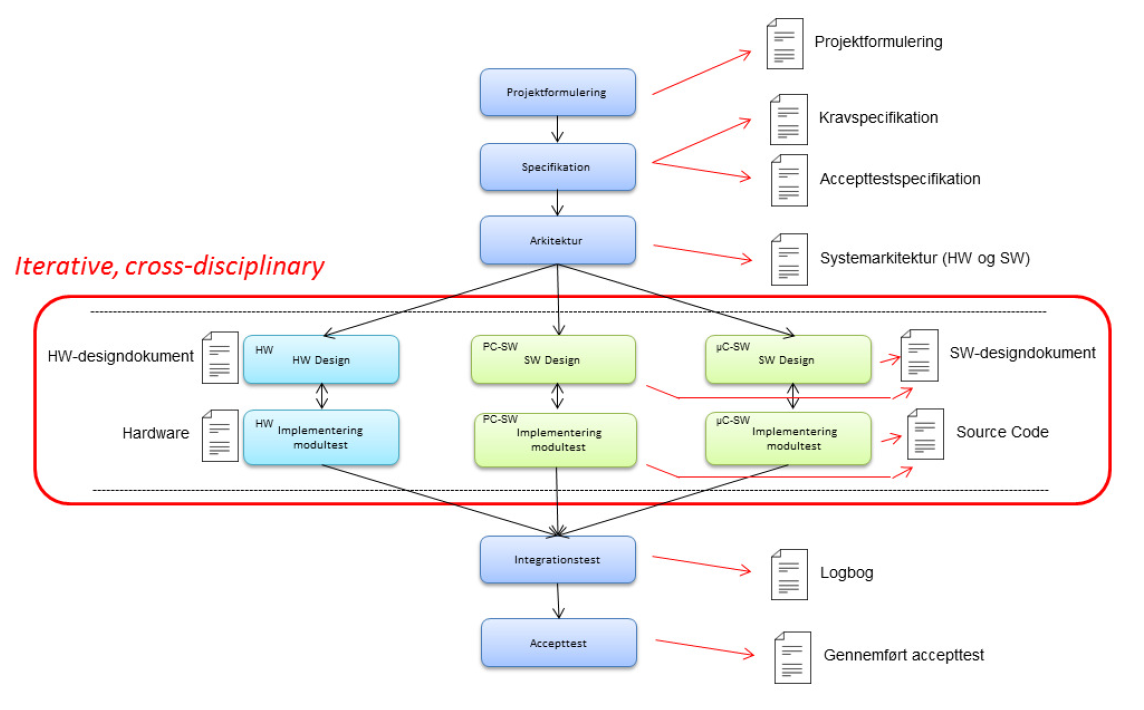
I de følgende afsnit vil projekts udførelse blive beskrevet.

## Projektgennemførelse (SN)

Dette projekt er gennemført som gruppearbejde.   
I projekts indledende faser blev i fællesskab udarbejdet en samarbejdskontrakt[[1]](#footnote-1), samt en overordnet filosofi for ambitionsniveauet i projekt gruppen. Der blev også aftalt at den interne kommunikation i projektgruppen skulle foregå via. ”*Slack*”, som kan forbindes både med google-kalender og github, således at al aktivitet i projektgruppen ville fremgå i kommunikations platformen.  
Der blev desuden aftalt at have en fast projektleder samt suppleant, og fast mødeleder samt suppleant, mens det blev aftalt at referent-rollen skulle gå på skift mellem gruppemedlemmerne.  
Projektlederens rolle har været at holde styr på projektets retning, og tidsplan, mens mødelederen rolle har været at sørge for mødeindkaldelser, dagsorden til møder, samt planlægning af møder med vejleder.

Stefan Nielsen har fungeret som projektleder, med Tonni Follmann som suppleant.  
Nikolai Topping har fungeret som mødeleder, med Anders Knudsen som suppleant.

Disse aftaler skulle således danne grundlag for det videre arbejde i projektgruppen.



Figur 1 - IHA´s udviklingsmodel.[[2]](#footnote-2)

Projektet er blevet udviklet med IHA´s udviklingsmodel for 2. semester projekt, som er vist på Figur 1.  
Denne udviklingsmodel indeholder følgende faser:

* Projektformuleringsfasen.
* Specifikationsfasen.
* Arkitekturfasen.
* Design- og implementeringsfasen.
* Accepttestfasen.

I de to første faser af udviklingen, arbejde hele projektgruppen sammen om at få det ønskede system specificeret.   
Herefter opdelte vi os i 3 mindre grupper, med selvfølgelig samarbejde mellem grupperne.  
Grupper er som følger:

* Software gruppe: Nikolai Topping, og Anders Knudsen.  
  Hovedansvar: At lave PC-Software.
* Styreboks gruppe: Tonni Follmann, Dennis Poulsen og Stefan Nielsen.  
  Hovedansvar: At lave hardware og software til styreboksen.
* Enheds gruppe: Martin Burmeister, Mikkel Espersen og Christian Bøeg-Jensen.  
  Hovedansvar: At lave hardware og software til enheder.

Arbejdet i de forskellige faser af udviklingen, beskrives yderligere videre i rapporten.

## Metode (SN)

I det følgende gives et overblik over de metoder som er anvendt i projektets specifikationsfase og arkitekturfase.

### Use cases

Vi har i specifikationsfasen anvendt use case analyse, som et primært værktøj til at bestemme systemets funktionelle krav.   
use case analysen tager udgangspunkt i projektformuleringen, som bruges til at identificere systemets primære- og sekundære aktører. Man ønsker så at identificerer alle de opgaver eller målsætninger som aktørerne ønsker systemet kan opfylde.   
Man laver så en use case som beskriver formålet med opgaven, hvordan systemet skal løse den opgave (hovedscenariet - set fra brugerens synspunkt), eventuelle afvigelser i forhold til hovedscenariet (extensions), samt forudsætninger for at use casen kan udføres.

### FURPS

Til at kortlægge systemets ikke-funktionelle krav (yderligere tekniske krav), blev primært anvendt analyse-metoden FURPS.  
FURPS er en forkortelse for ”Functionality, Usability, Reliability, Performance and Support”.  
Ved at gennemgå de 5 kategorier en efter en, og stille spørgsmål til hvad vores system burde opfylde at krav inden for den pågældende kategori, kan man således identificerer ikke-funktionelle krav for systemet, som ikke kom frem under use case analysen.

### SysML

I arkitekturfasen er desuden anvendt SysML værktøjerne:

#### BDD – Block Definition Diagram

Når man laver BDD for et system, starter man ud med at opdele systemet i mindre hardware-blokke, som er nødvendige for at opfylde systemets krav.   
Med BDD´er kommer man frem til et visuelt overblik over systemets del-komponenter, og hvilke del-komponenter der arbejder sammen, og dermed må have forbindelse til hinanden.

#### IBD – Internal Block Diagram

IBD´er er brugt til at danne overblik over de forskellige blokkes forbindelser, og signaltyper.

### UML

I projektets arkitekturfase har vi anvendt Domæneanalyse til at identificerer systemets klasser, deres indbyrdes relationer, og klassernes individuelle funktioner.   
Man starter med at gennemgå use cases for systemet, og gennem navneords analyse identificerer konceptuelle klasser, og så bestemme om de identificerede klasser er boundary- domæne- eller kontrolklasser.

#### SD – Sequence Diagram

Vi gik så videre til at lave sekvensdiagrammer for alle CPU´er i systemet. Der blev først udfærdiget en matrice, som viste hvilken CPU (styreboks, enhed eller PC) der var en del af hver use case, og så blev der lavet sekvensdiagrammer for hver CPU, for alle de use cases som den CPU indgik i, og deres udvidelser.

Sekvens diagrammerne hjælper med at identificerer funktioner, som så kan indsættes i de klasser som er identificeret ved domæne analysen, og på den måde kan man komme frem til et samlet (indledende) UML klasse diagram for systemet.

1. Se bilag: ”Samarbejdskontrakt” [↑](#footnote-ref-1)
2. Se bilag: Vejledning for gennemførelse af projekt 2.pdf [↑](#footnote-ref-2)