Bacheloroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

Eivind Andreas Erdahl Kaspersen
Tobias Rengård

Concurrent Design og Programvareutvikling

"Hvordan egner Concurrent Design (CCD) seg som prosjektmetode i utvikling av programvare?"

Bacheloroppgave i Digital Forretningsutvikling
Veileder: Torstein Elias Løland Hjelle
Mai 2020

NTNU
Kunnskap for en bedre verden

Eivind Andreas Erdahl Kaspersen
Tobias Rengård

Concurrent Design og Programvareutvikling

"Hvordan egner Concurrent Design (CCD) seg som prosjektmetode i utvikling av programvare?"

Bacheloroppgave i Digital Forretningsutvikling
Veileder: Torstein Elias Løland Hjelle
Mai 2020

Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

NTNU
Kunnskap for en bedre verden

Sammendrag

Hensikten med denne oppgaven er å få et innblikk i, og kartlegge hvordan utviklingen av programvare foregår i dag og om metoden Concurrent Design (strukturert metode basert på tverrfaglig samarbeid i sanntid) kan tilføre noe av verdi på dette området. Problemstillingen vi har brukt lyder som følger: *"Hvordan egner Concurrent Design (CCD) seg som prosjektmetode i utvikling av programvare?".*

Resultatene av denne studien bygger på et teoretisk grunnlag av programvareutvikling og CCD, etterfulgt av syv kvalitative semistrukturerte dybdeintervjuer med en representativ gruppe sentrale deltakere i prosessen innen utvikling av digitale produkter og tjenester. Disse er hovedsakelig ressurser knyttet til konsulentvirksomhet og med bakgrunn fra forskjellige sektorer (privat og offentlig).

Konklusjonen vår fra dette arbeidet viser at såkalt smidige utviklingsmetoder i stor grad foretrekkes, og er den "de facto" fremgangsmåten for utviklingen av programvare i dag. Det er en del fellesnevnere mellom arbeidsprosessene for CCD og utviklingen av programvare, som for eksempel workshops. CCD er imidlertid karakterisert av en mer rigid og systematisk struktur i forhold til det vi ser i de smidige metodene. Vår oppfatning er at CCD kan bli for omfattende til direkte bruk i programvareutviklingsprosjekter i forhold til dagens praksis, som har vist seg å dra nytte av mer fleksibilitet.

Summary

The purpose of this thesis is to gain insight and to outline the current situation of software development and to assess if the Concurrent Design method (structured method based on interdisciplinary teamwork in real time) can add value in this area. The research question we have constructed is as follows: "In what way is Concurrent Design (CCD) suitable as a project method in software development?".

The results of our study rests upon the theoretical foundation of software development and CCD, followed by a total of seven semi-structured interviews with a representative group of central participants of the process within the development of digital products and services. Our respondents are mainly related to consultancy and with backgrounds from different sectors (private and public).

Our conclusion shows that so-called flexible development methods are much preferred and is the "de facto" approach to software development today. There are some common denominators between the CCD work processes and software development, such as workshops. However, CCD is characterized by a more rigid and systematic structure compared to what we see in the flexible methods. Our view is that CCD may become too extensive for direct use in software development projects in relation to current practice, which has proven to benefit from more flexibility.

Innhold

Sammendrag 1
Summary 1
Innledning 4
Bakgrunn for valg av tema 4
Problemstilling/Hensikt 4
Om oppgavens struktur 5
Teori 6
Programvareutviklingsmetoder 6
Concurrent Design (CCD) 9
Vår forståelse av teorien 12
Metode 13
Forskningsdesign 13
Kvalitativt intervju 13
Valg av informanter (utvalg) 14
Intervjuguide 14
Gjennomføring av intervju 14
Behandling av data 15
Etikk 15
Kvalitet 16
Analyse 17
Prosjektmetoder i programvareutvikling 17
Mennesker 19
Prosesser 20
Verktøy 21
Drøfting 23
Avslutning 27
Kritikk 28
Kilder 29
Vedlegg 31

Innledning

Bakgrunn for valg av tema

Gjennom denne oppgaven ønsker vi å undersøke hvor egnet Concurrent Design metoden er for produktutvikling av digitale produkter og tjenester. Dette vil vi gjøre gjennom å se på hvordan ulike bedrifter går frem og hvilke metoder de benytter. Vi vil sette disse erfaringene, samt eksisterende teori på feltet opp imot de teoretiske egenskapene til Concurrent Design metoden for å finne ut om det er noen hull i dagens fremgangsmåter som Concurrent Design potensielt kan fylle.

Bakgrunnen for valg av tema er enkel; i utdannelsen "Digital Forretningsutvikling" står tjeneste- og produktutvikling sentralt, spesielt utviklingen av digitale tjenester og produkter. Gjennom studiet fikk vi også teoretisk kunnskap knyttet til metoden Concurrent Design, samt litt praktisk erfaring med denne. Vår erfaring med metoden var positiv og fikk begge inntrykket av at dette var en metode som kunne gi økt effektivitet, samtidig som at den gav et solid grunnlag for bedre samarbeid på tvers av ulike team.

Når det etterhvert kom til valg av bacheloroppgave ble vi nysgjerrige på om Concurrent Designs egenskaper har en overføringsverdi til utviklingen av digitale produkter, da den tross alt baserer seg på metoden Concurrent Engineering, en metode utviklet for produktutvikling med fokus på å sparte kostnader gjennom kortere ledetid (Sohlenius, 1992).

Problemstilling/Hensikt

Hensikten med denne oppgaven er å få et innblikk i, og kartlegge hvordan utviklingen av programvare foregår i dag og om metoden Concurrent Design kan tilføre noe av verdi på dette området. Vår antagelse er at produkt- og tjenesteutvikling, uansett form, får best resultater gjennom tverrfaglig samhandling. Vi ser også for oss at et annet nøkkelkriterie for å oppnå suksess innenfor produktutvikling generelt er effektivitet, både for å spare kostnader og for å nå markedet tidlig. Concurrent Design, slik vi har blitt kjent med den, er en metode vi ser for oss at skal kunne utmerke seg på begge disse områdene.

Problemstillingen vi har basert vårt arbeid på, lyder som følger:

«Hvordan egner Concurrent Design (CCD) seg som prosjektmetode i utviklingen av programvare?»

Videre ønsker vi gjennom forskningsspørsmålene å skape et bilde av dagens metode-situasjon innen programvareutvikling samt å kunne presisere og forsterke fokuset på CCD sin rolle i dette bildet. Forskningsspørsmålene er som følger:

«Hvilke programvareutviklingsmetoder benyttes i dag og hvilke krav stilles til dem?»

«Hvordan anvendes dagens programvareutviklingsmetoder?»

«I hvilken grad samsvarer dagens praksis med prinsippene anvendt i CCD?»

«Hva kan CCD tilføre av verdi inn i dagens praksis?»

Om oppgavens struktur

Oppgaven er bygget opp på generelle empiriske prinsipper, med en kapittelstruktur som tar utgangspunkt i empiriske oppgaver. Oppgaven består av seks kapitler. Innledningsvis, i kapittel presenteres bakgrunnen for oppgaven og hvorfor denne er valgt, samt en presentasjon av problemstillingen. I kapittel to vil vi ta for oss relevant teori for å introdusere leser for teorigrunnlaget vi går ut ifra. Gjennom kapittel tre vil leseren få innsikt i den metodiske fremgangsmåten som er benyttet og hvorfor denne er valgt. Siste halvdel av oppgaven består av de avsluttende kapitlene analyse, drøfting og avslutning. I analysekapittelet vil vi presentere funnene våre før vi i kapittel fem vil drøfte disse funnene i sammenheng med problemstilling og eksisterende teori. Det avsluttende kapittelet vil presentere vår konklusjon.

Teori

Programvareutviklingsmetoder

New product development (NPD) var lenge en utviklingsprosess med en «relay»-type tilnærming hvor én spesialistgruppe i prosjektet overtok stafettpinnen etter en annen gjennom hele utviklingens livsløp (Takeuchi & Nonaka, 1986). Denne metoden for NPD så et skifte på 1980-tallet ettersom flere bedrifter så et krav fra markedet om hurtighet og fleksibilitet. Som et følge av dette oppstod uttrykket «relay to rugby» som blir beskrevet av Takeuchi og Nonaka i «The New New Product Development» (Takeuchi & Nonaka, 1986). Gjennom «Rugby»-tilnærmingen ble produktutviklingsprosessen forandret fra et strukturert løp med avgrensede faser og tilhørende spesialistgrupper, til et dynamisk løp med overlappende prosjektfaser og et samspill mellom spesialistgruppene. I årene som kom så man opphav til ulike metoder med trekk fra «rugby»-metoden, hvor kryssfunksjonelle utviklingsteam og «concurrent engineering» (CCE) var noen av de mest utspredte metodene (Smith & Reinertsen, 1992). Smith og Reinertsen forklarer at som en følge av adopsjonen til disse metodene hos et flertall av bedrifter, forsvant også konkurransefordelene som fulgte med dem. «*Decisive advantage is likely to come from the techniques competitors are not using*», skrev Smith og Reinertsen, og stilte med det krav til å utforske nye teknikker for å skape konkurransefortrinn.

Som et resultat av den økende digitaliseringen på starten av 2000-tallet så man en utvikling som utforsket ukjente områder og det oppstod usikkerhet rundt hvordan en skulle tilnærme seg utviklingsprosjektene bak programvare (Dingsøyr, Nerur, Balijepally, & Moe, 2012). Gjennom digitale verktøy og et dynamisk marked hvor endringer skjedde raskt, skaptes en oppfattelse om at utviklingen av digitale produkter ville stille krav til andre suksessfaktorer enn «tradisjonell» produktutvikling og NPD (Banfield, Lombardo, & Wax, 2015). Det ble derimot argumentert for at forskjellene lå i hvor fokuset til de to industriene var plassert og ikke i utfordringene eller kravene en kom over (Nambisan & Wilemon, 2000). Likevel så en at hvor tradisjonell produktutvikling er preget av tydelig definerte prosjekter med godt dokumenterte og forankrede egenskaper, funksjoner og krav (Fernandez & Fernandez, 2008), kunne utviklingsprosjekter knyttet til programvare dra fordel av en annen tilnærming (Dingsøyr et al., 2012). Som et følge av dette oppstod det også en nødvendighet for utforskning av nye metodeverk innen programvareutviklingen (Abrahamsson, Salo, Ronkainen, & Warsta, 2017).

Nye krav fra den voksende digitale industrien ville etter hvert føre til oppstanden av flere innovative metodikker. På starten av 2000-tallet ble det tydelig at tradisjonelle utviklingsmetoder av digitale produkter som tjenester og programvare, ikke tilstrekkelig klarte å holde tritt med den raskt endrende og turbulente programvareindustrien i internet-æraen (Highsmith & Highsmith, 2002). Flere nye "smidige metoder" («Agile methods») ble skapt som en respons til dette, og på grunn av kompatibiliteten til programvareindustriens behov, egenskaper og rutiner rundt digitale verktøy, ville de vise seg å dominere utover 2000-tallet (Dingsøyr et al., 2012). I boken "Agile Software Development Methods: Review and Analysis" undersøker Abrahamsson et al. hva en smidig metode er, hva den går ut på og hvorfor utvikling av digitale tjenester og programvare domineres av denne metodikken (Abrahamsson et al., 2017). De argumenterer for at de tradisjonelle plandrevne metodene er for mekaniske og at konkurransedyktige metoder må være fleksible og dynamiske. De forklarer også at adopsjonen av smidige metoder har skapt en industri hvor nye metoder oppstår regelmessig som igjen har skapt mer forvirring enn klarhet hos aktørene. Hver metode bruker sitt eget sett av vokabular og terminologier og en overgang fra en metode til en annen kan virke avskrekkende. De smidige metodene får også kritikk for å ha problemer knyttet til oppskalering (Vilkki, 2010) selv om motstridende funn er gjort i andre undersøkelser (Jørgensen, 2018). Videre stiller smidige metoder krav til erfarne teammedlemmer, som kan være med på å øke sjansen for suksess uavhengig av metode (Fernandez & Fernandez, 2008).

I nyere tid har den smidige metodikken vært ledende innen utvikling av digitale produkter og tjenester (Abrahamsson et al., 2017). I følge Stober & Hansmann vil metodene som tar i bruk den smidige filosofien legge vekt på en horisontal tankegang som krysser rammesett som team, geografi og organisasjoner, og skape et skifte både i arbeidskulturen til organisasjonen og innstillingen til deltakerne (Stober & Hansmann, 2010). Videre beskrives det at smidige løsninger vil fokusere på endring og kreve at bedriften skaper et dynamisk og responsivt arbeidsmiljø. Kundens verdi og inkludering understrekes også, samt opprettholdelse av en bærekraftig utviklingshastighet. I praktisk forstand kan de smidige metodene være bestående av inkrementell og iterasjons arbeid, samarbeid i team tvers av fagfelt, enkle og forståelige metoder, og tilpasningsdyktighet (Abrahamsson et al., 2017). Flere metoder betegnes nå som å ligge under paraplyen til den smidige metodikken. Blant de største finner en Extreme

Programming, Scrum, Crystal, Feature Driven Developmend (FDD), og Rational Unified Process (RUP) for å nevne noen.

Scrum metodikken er en av de vanligste metodene og blir brukt globalt på flere små og store prosjekter og i alt fra komplekse til enklere løsninger (Permana, 2015). Permana beskriver den som å ha et sterkt fokus på strategi og å skape en fleksibel og holistisk arbeidskultur hvor et team jobber sammen for å oppnå mål (Permana, 2015). Videre er det skrevet at metoden er oppbygd av kontinuerlig testing og iterasjonsarbeid for å alltid kunne være responsiv til nye krav til funksjonalitet eller design (Woodward, Surdek, & Ganis, 2010). Rent praktisk er Scrum avhengig av en inndeling i roller (produkteier, Scrum master og Team) samt et fokus på gjennomførelsen av spesifikke prosesser som "sprints" og "standup-møter" (Woodward et al., 2010).

Mellom de smidige metodene og det tradisjonelle metodeverket eksisterer det grunnleggende forskjeller, likevel kan disse ulikhetene legge til rette for hybrid-løsninger hvor bruk av begge metodeverkene vil komplimentere hverandre (Fernandez & Fernandez, 2008). Dette argumentet trekker linjer til en tankegang rundt det å finne en balanse mellom «struktur og kaos» som beskrevet av Cunha og Gomes om den generelle produktutviklingsprosessen (Cunha & Gomes, 2003). Med bakgrunn i egenskapene til de to metodeverkene kan en se på de tradisjonelle metodene som «struktur» og de smidige som «kaos» og ut ifra en slik symbiose har en mulighet til å oppnå ulike fordeler (Fernandez & Fernandez, 2008). Utfordringen er deretter å finne ut i hvilken grad en skal bruke hver av de ulike metodeverkene. I en undersøkelse som stilte spørsmål ved egnetheten til å blande tradisjonelle og smidige metoder i et prosjekt, fant Špundak at det er en rekke karakteristikker som er avgjørende (Špundak, 2014). Prosjektstørrelse, grad av risiko for systemkritiske hendelser, og linearitet av prosjektplan var noen av disse. For å bestemme i hvilket forhold en skal blande de to ulike metodene kreves det innsikt i sammensetningen av disse egenskapene hos prosjektet som gjør at en kan skape en egendefinert metodekomposisjon.

Med bakgrunn i den økende graden av komplekse og usikre prosjektsituasjoner i nyere tid (Fernandez & Fernandez, 2008) kan det argumenteres at programvareutviklingsprosjekter krever et bredt utvalg av metoder eller i det minste en mulighet til å tilpasse metodologien til et spesifikt prosjekt (Cockburn, 2006). «Agile Tailoring» er en betegnelse på en slik prosess som

omfatter å velge ut egenskaper og prosesser fra ulike metoder for å skreddersy og skape en metode til sine prosjekter (Conboy & Fitzgerald, 2007). Målet med denne prosessen er å bedre svare på både forretningsmessige og prosjektrelaterte krav hos bedriften (Campanelli & Parreiras, 2015). Om denne måten å behandle metoder på er gunstig kan en se uenighet i (Conboy & Fitzgerald, 2007). I sin undersøkelse forteller Conboy og Fitzgerald at smidige metoder ikke er formbare selv om dette undergraver hele konsepter rundt å være smidig. Likevel beskriver annen litteratur metoder på hvordan en kan dra utnytte av «Agile Tailoring» (Campanelli & Parreiras, 2015) blant annet i komplekse programmer med høy risiko for systemsvikt (Bass, 2016).

Concurrent Design (CCD)

Concurrent Design er en prosjektledelsesmetodikk nært knyttet til Concurrent Engineering. Concurrent Engineering er 9efiner slik: "Concurrent Engineering is a systematic approach to the integrated, concurrent design of products and their related processes, including manufacture and support. This approach is intended to cause the developers, from the outset, to consider all elements of the product life cycle conception through disposal, including quality, cost, schedule, and user requirements" (Winner, Pennell, Bertrand, & Slusarczuk, 1988). Concurrent Engineering legger stor vekt på det faktum at en skal jobbe tverrfaglig, sammen og samtidig, fra et tidlig stadium i prosjektet slik at en får integrert kunnskap og kompetanse.

Dr. Knut I. Oxnevad tok dette konseptet videre og utviklet det han omtaler som Concurrent Design. Oxnevad jobbet ved NASA Langley Jet Propulsion Laboratory da han sammen med sitt team utviklet denne nye tilnærmingen til utviklingen av romfartøy og som Oxnevad gav tilnavnet Concurrent Design. Oxnevad var en av de første til å bruke dette begrepet om metoden og omtales i dag som en av pionerene og en anerkjent foregangsfigur for Concurrent Design. I 2000 publiserte han "The eight principles of Concurrent Design" (SIMTANO, 2020b). I tillegg til disse åtte prinsippene er "People, process and Tool" (PPT) modellen en av pilarene for Concurrent Design. (SIMTANO, 2020a).

Dr. Knut I. Oxnevad er grunnlegger og i dag administrerende direktør for Simtano, et selskap som benytter de åtte prinsippene for Concurrent Design samt "People, Process and Tools"-modellen for å forbedre arbeidsprosessene til sine bedriftskunder.

For denne oppgaven tar vi utgangspunkt i Oxnevads (Oxnevad, 2000) beskrivelse av Concurrent Design, som bygger på de to pilarene: De åtte prinsippene for Concurrent Design og treenigheten mellom mennesker, prosesser og verktøy (PPT-modellen). Den første pilaren er bygget på følgende sentrale prinsipper: (1) Analyse- og designaktiviteter er utført av tverrfaglige designteam; (2) medlemmene i designteamene jobber sammen i samtidige sesjoner; (3) "kunder" og teammedlemmer deltar i samtidige sesjoner; (4) analyse- og designaktiviteter skjer samtidig, i sanntid; (5) sammenkoblede "high-end" dataverktøy brukes i de samtidige sesjonene; (6) disse dataverktøyene skal være i bruk fra tidlig i design syklusen; (7) vanlig geometriske data (CAD) deles elektronisk mellom dataverktøyene; (8) CAD, strukturell, termisk og optiske data kan eksporteres til og fra designteamet.

De åtte prinsippene Oxnevad legger frem vil en sikre en total systemtilnærming gjennom tverrfaglige team som innehar kunnskap og kompetanse innenfor alle relevante fagfelt. Kundedeltakelse vil sikre at alle valg og beslutninger som blir tatt vil kunne utfordres i sanntid, og endringer gjøres der det sees nødvendig. Fordi dette skjer i sanntid vil også valg kunne begrunnes og kunden overbevises der det finnes usikkerhet.

Sesjonene som gjennomføres i Concurrent Design bærer preg av at faktisk arbeid skal gjennomføres. Gjennom å ha tilgang på høyteknologiske verktøy vil deltakerne sammen kunne gå til verks på selve oppgaven, til forskjell fra mer tradisjonelle møter der en hovedsakelig utveksler informasjon. I sesjonslokalet skal det legges opp til at en en har verktøy for sømløs overføring og deling av design og ulike typer data på tvers av kompetansegruppene i standardiserte grensesnitt.

Det er viktig å komme i gang med tilgjengelige verktøy i en tidlig fase slik at en deltagerne kan gjennomføre konkrete oppgaver slik at det tidlige i utviklingsfasen utvkles design og planer som videre kan raffineres.

Gjennom delingen av data vil design fra en tidlig fase være tilgjengelig for bearbeiding i neste fase, og designteamet kan være hand-on support. Som følge av dette kan designfasen sees på som én prosess, kontra flere prosesser som er lenket sammen (Oxnevad, 2000).

Den andre pilaren bygger på mennesker, prosesser og verktøy (PPT-modellen). Modellen forutsetter at det er gjensidig samspill mellom denne treenigheten. Concurrent Design handler i stor grad om å benytte PPT-modellen på en mer strukturert måte for å kunne fatte bedre og

raskere beslutninger (Oxnevad, 2000). I Concurrent Design kan vi beskrive PPT-modellens innhold som følger:

- Mennesker – I Concurrent Design kan vi dele deltakerne inn i fem ulike roller. Prosjektleder, fasilitator, sesjonssekretær, fageksperter og øvrige roller (kunde etc.). Prosjektlederen har som rolle å lede prosjektet, men overlater styringen av selve sesjonene til fasilitatoren. Prosjektlederen vil i sesjonene utøve prosjektadministrativt arbeid. Fasilitatoren er en tilrettelegger i sesjonene og skal sørge for at gruppearbeidet går smidig gjennom å stimulere til arbeidet og kommunikasjon blant gruppene. En viktig faktor vil være å sikre et godt arbeidsmiljø og motivere deltakerne gjennom sesjonen, samt og tørre å "tvinge" frem beslutninger dersom arbeidet stopper opp ved et veivalg. Som fasilitatorene høyrehånd har vi sesjonssekretæren som skal sørge for at arbeidsrommet er oppe og går rent teknisk, samt skrive sesjonsreferat og notere aksjoner og beslutninger som blir tatt. Hvilke fageksperter som behøves avhenger av typen prosjekt, men det er svært viktig at det er fagekspert nok til å dekke alle fagdisipliner som er nødvendige for å nå målet for prosjektet. Fagekspertene **skal** ha myndighet til å kunne ta endelige beslutninger for valg innen deres fagfelt.
- Prosesser - ISO 9000 definisjonen av en prosess er som følger: "Samling av beslektede eller samvirkende aktiviteter som omformer tilført grunnlag til resultater" (ISO, 2018). I tilfellet til Concurrent Design handler dette hovedsakelig om sesjonene som gjennomføres – det er disse som skille Concurrent Design fra andre prosjektmetoder. Sesjonene har en varighet på mellom 3-5 timer og består av intensive tverrfaglige arbeidsøkter som sammen med "vanlig" arbeid mellom sesjonene skal resultere i ønsket resultat.
- Verktøy – Disse vil avhenge av prosjektets natur og mål da forskjellige eksperter vil ha behov for ulike ekspertverktøy, men det vil være en del felles verktøy for alle Concurrent Design sesjoner. Samtlige sesjoner vi ha behov for et anlegg der en kan gjennomføre sesjoner. Anlegget må ha installert verktøy som legger til rette for samhandling og samarbeid, slik som arbeidsstasjoner med installert programvare som gjør det mulig for enhver å prosjektere sitt skjerminnhold til over til en eller flere

prosjektorer installert i rommet. Dette skal bidra til å forbedre interaksjonen mellom de deltakende gruppene (Oxnevad, 2000).

Vår forståelse av teorien

I gjennomgangen av teori har vi sett på overgangen fra produktutvikling til programvareutvikling og modningen av prosjektmetoder de siste 30 årene. Helt fra «relay-to-rugby» tilnærming på 80-tallet og frem til bruken av smidige metoder som scrum i dag. Vi ser at dagens situasjon viser en betydelig adopsjon av de smidige metodeverkene ettersom de svarer godt på kravene til industrien tilknyttet evne til å endres, responsivitet og å være dynamisk. Teorien forteller også at programvareindustrien i større grad er egnet for slike krav ettersom industrien har eksisterende rutiner rundt digitale verktøy. De nevnte kravene så vi komme av at industrien kontinuerlig opplever skifter og innovasjon som vil kunne påvirke virksomheter internt. Videre er det vår forståelse at smidige metoder i dagens situasjon er preget av egendefinert bruk og sammensetning av metoder. Av de smidige metodene har vi funnet at scrum er blant de mest populære og utbredte.

CCD kjennetegnes av en prosess der et multidisiplinært team anvender en integrert designmodell ved bruk av dedikerte fasiliteter og infrastruktur, og er rammeverket for utvikling av teamet for interdisiplinært samarbeid mellom deltakere i sanntid. Disse deltakerne har egenskaper som omfatter teknisk og funksjonell ekspertise i tillegg til å være løsningsorienterte og beslutningsdyktige samt at der bør inneha samarbeidsegenskaper. Teamet består typisk av fasilitator, sekretær, domene eksperter, prosjektleder og kunderepresentanter som er gjensidig ansvarlige, og besitter komplementære egenskaper for å kunne produserer solide løsninger. Oppbygningen av CCD baserer seg på de to pillarene; treenigheten (PPT) av mennesker, prosesser og verktøy, og de 8 prinsippene. Vår forståelse er at et videre utgangspunkt i treenigheten vil skape et bedre bilde av hvordan CCD egner seg som prosjektmetode i programvareutvikling ettersom de 8 prinsippene i større grad beskriver prosesser i allerede etablerte CCD-prosjekter men at treenigheten derimot legger et krav til grunnleggende egenskaper ved prosjektet.

Metode

Gjennom dette kapitlet vil vi presenterer valg av forskningsmetode og hvorfor vi har gått for disse valgene. Hensikten med dette er å gjøre oppgaven som transparent som mulig. Vi vil belyse, og forklare, hvordan vi har gått frem som forskere i undersøkelsen gjennom å beskrive prosessen og arbeidet med datamaterialet, samt hvordan vi har valgt våre informanter.

Forskningsdesign

Vi ønsker med oppgaven å få et innblikk i hvordan Concurrent Design metoden kan bidra til å forbedre/effektivisere produktutviklingen av digitale produkter og tjenester. Når det kommer til valg av forskningsdesign fant vi fram til tilnærmingen *stegvis-deduktiv induktiv metode,* som forfatteren av metoden beskriver som en metode med "nysgjerrighet som utgangspunkt og generaliserbar forståelse som mål" (Tjora, 2012, p. 16). Denne beskrivelsen passet godt med våre ambisjoner og hvordan vi så for oss at forskningen burde foregå.

Med utgangspunkt i en induktiv tilnærming vil vi jobbe eksplorerende og empiridrevet, noe som gir oss muligheten til å jobbe med egen empiri og sette den opp imot eksisterende teori (Tjora, 2012, p. 18).

Dette forskningsdesignet er av en kvalitativ art. Vi valgte å gå denne retningen fordi vi antok at den tid og ressurser en bacheloroppgave bevilger ikke ville gitt oss nok rådata til å gi et solid nok fundament for en god kvantitativ avhandling. Dette på bakgrunn av at vi ikke ville rukket å få implementert CCD inn i en produktutviklingsprosess.

Kvalitativt intervju

Innenfor den kvalitative forskningsmetoden er intervjuer den mest benyttede metoden for innsamling av empiri. Som følge av at vi har gått for et eksplorerende forskningsdesign har vi valgt å benytte oss av semi-strukturerte dybdeintervjuer for innhenting av egen empiri. Semistrukturerte intervjuer er intervjuer med en halvfast struktur (Tjora, 2012, p. 229), noe som gir oss større fleksibilitet i forhold til å stille oppfølgingsspørsmål. Oppfølgingsspørsmål gir oss muligheten til å grave videre inn i nyoppdaget materie som skulle dukke opp i intervjuene.

En svakhet ved denne typen intervjuer er at vår erfaring innen gjennomføring av intervjuer potensielt vil kunne ha stor innvirkning på intervjuets kvalitet og dermed påvirke kvaliteten på empirien. Vår manglende erfaring kan føre til at vi stiller dårlig formulerte spørsmål, ledende

spørsmål eller at vi ikke har kunnskap og kompetanse nok til å stille de rette oppfølgingsspørsmålene.

Valg av informanter (utvalg)

Med et ønske om å finne ut av hvordan CCD-metoden kan bidra i produktutvikling ønsket vi å få innsikt gjennom å ta kontakt med, og intervjue to grupper med informanter – en gruppe med erfaring og ekspertise innen CCD og en annen med erfaring og ekspertise innen produktutvikling. Aller helst skulle vi intervjuet informanter med erfaring på begge områder, men det viste seg vanskelig å finne.

Vi kartla ulike bedrifter som hadde erfaring med produktutvikling eller CCD for så å sende forespørsler om intervjuer til disse. Vi benyttet videre *snøballmetoden* for å komme i kontakt med flere relevante informanter. Snøballmetoden går ut på å rekruttere nye deltakere til undersøkelse ved å spørre allerede påmeldte deltakere. Ved å benytte denne metoden for rekruttering står en i fare for å begrense variasjonen i utvalget (Tjora, 2012, p. 230).

Intervjuguide

Som nevnt så har vi valgt å benytte oss av en semistrukturert tilnærming til intervjuene våre. Dette betyr at intervjuguiden fungerer som nettopp en guide. Vi har på forhånd laget en liste med spørsmål vi ønsker å få svar på, men vel så viktig er det at vi som forskere tillater oss å være en aktiv del av samtalen og spørre og grave i interessante og relevante emner som dukker opp, og som vi ikke på forhånd hadde skrevet ned. Vi hadde på forhånd, i tillegg til de ferdig formulerte spørsmålene, også satt opp stikkord som inspirasjon til mulige oppfølgingsspørsmål tilknyttet spørsmålene slik at vi hadde disse å lene oss på under intervjuet. Dette var kritisk for oss som "ferske" intervjuere med lite erfaring.

Gjennomføring av intervju

Grunnet koronasituasjonen vi befinner oss i, ble alle intervjuer gjort via videokonferanser. "Dybdeintervjuer betraktes som intervjuer som skjer ansikt-til-ansikt mellom informant og intervjuer" skriver (Tjora, 2012, p. 140), som påpeker at kroppsspråk og det "å nikke for å få informanten til å snakke videre i den retningen vedkommende er" er sentrale aspekter av det som gjør dybdeintervju til et dybdeintervju. Vi følte derfor at de videokonferanse ble den beste måten å løse intervjusituasjonen på slik situasjonen ble.

På forhånd av intervjuene sendte vi mail med informasjon om CCD til informantene slik at de skulle få noe tid til å se på hva dette er for noe og eventuelt ha anledningen til å komme opp med spørsmål til oss om det skulle være noe som var utydelig.

Intervjuene ble startet med litt løs prat der vi introduserte oss selv, temaet og hvorfor vi ønsket se nærmere på dette. Gjennom den introduserende praten forhørte vi oss også om informanten samtykket til lydopptak av samtalen. Vi gav deretter informanten på nytt en kortfattet gjennomgang i hva Concurrent Design er med anledning til å stille spørsmål basert på e-mailen de allerede hadde fått, før vi begynte på spørsmålene våre. Vi startet med å snakke om deres arbeidserfaringer og mer spesifikt om deres erfaring med produktutvikling av digitale produkter og hvordan dette gjennomføres på deres arbeidsplass. Vi fulgte deretter opp med mer konkrete spørsmål som vi kunne benytte til å sette metodikken deres opp mot metodikken i Concurrent Design.

Intervjuene ble tatt opp på lydopptaker slik at vi kunne ha fullt fokus på samtalen, noe vi så på som svært viktig da vi skulle komme opp med både spørsmål og oppfølgingsspørsmål, noe som kunne blitt vanskelig om vi måtte dokumentere og notere under intervjuene. Disse lydopptakene ble så transkribert gjennom programvaren "Transcribe" fra Wreally Studios.

Behandling av data

Etter at intervjuene var gjennomført og transkribert satt vi igjen med både lydfiler og tekstfiler. Samtlige av disse ble lagret i skyen via Google Drive, slik at vi begge hadde tilgang på disse for videre analyse. Etter at analysen er ferdig vil filene slettes, senest innen fristen den 6. juni 2020.

Etikk

Etikken knyttet til forskning er generelt sett mest opptatt av at informantene ikke skal komme til skade (Tjora, 2012, p. 159). Det første steget rent etisk er å informere informantene om hvordan dataene skal brukes, hvordan materialet skal lagres og hva som skjer med materialet når forskningen er ferdiggjort. Deretter, når informanten er klar over hva det er en melder seg opp til, er å passe på at informanten frivillig samtykker til å være med i undersøkelsen. Det må også før intervjuet er i gang avklares hvorvidt lydopptak er greit, og om det er noen spesielle betingelser knyttet til dette – ønsker informant å være anonym?

I vårt tilfelle gav alle informanter sitt samtykke til taleopptak, og det var ingen betingelser rundt anonymitet eller spesiell behandling av dataen. Vi ser det ikke som sannsynlig at noe av datamaterialet vi samler inn kan klassifiseres som sensitivt eller at noen vil "føle seg tråkket på tærne".

Kvalitet

Vi må ha kvalitet på forskningen vår. De mest brukte indikatorene på kvalitet er begrepene reliabilitet, validitet og generaliserbarhet. Reliabilitet og validitet oversettes gjerne på norsk til henholdsvis pålitelighet og gyldighet (Tjora, 2012, p. 202).

Når det kommer til pålitelighet hevdes det innenfor den positivistiske posisjon at nøytralitet og en objektiv observatør er idealet (Tjora, 2012, p. 203). I så måte kan en personlig interesse og forkunnskap om emnet være en forstyrrelse. I forhold til personlig interesse hos oss som forskere finnes denne kun i form av nysgjerrighet og resultatet vil for oss være interessant, uansett utfall. For å kommunisere en objektiv fremgangsmåte med leseren vil vi tydeliggjøre hva som er kommer direkte fra datagenerering og hva som er et produkt av analyse, slik Clive Seale påpeker er viktig (Seale, 1999).

Gyldigheten av oppgaven handler hovedsakelig om at de svarene vi får gjennom forskningen svarer på de spørsmålene vi faktisk har stilt oss (Tjora, 2012, p. 203). For å sikre oss på dette punktet, og for at leseren skal være trygg det at det som står faktisk stemmer, må vi åpne opp om hvordan vi går fram i forskningen og hvorfor vi gjør det på den måten vi gjør. Til slutt er det et viktig poeng at vi som forskere passer på at vår forskning er forankret i annen relevant forskning, gjennom blant annet god kildehenvisning.

For generaliserbarhet stiller vi nok relativt svakt med et utvalg på syv personer, men dette er som forventet med tanke på at vi har gått for en kvalitativ tilnærming med de restriksjoner en 20 studiepoengs bacheloroppgave gir. Til tross for at generaliserbarheten er svak, vil oppgaven gi oss en indikator, og vil kunne være et utgangspunkt for videre forskning.

Analyse

Dette analysekapittelet bygger på observasjoner fra intervjuer med en representativ gruppe sentrale deltakere i prosessen innen utvikling av digitale produkter og tjenester. Disse er hovedsakelig ressurser knyttet til konsulentvirksomhet og med bakgrunn fra forskjellige sektorer (privat og offentlig).

Hensikten er å presentere en strukturert oversikt over elementene anvendt i arbeidet med å utvikle slike produkter og tjenester, dette som en basis for å drøfte i hvilken grad dagens praksis samsvarer med prinsippene anvendt for Concurrent Design (CCD) og hvorvidt CCD kan tilføre noe av verdi inn i dagens praksis.

En oppsummering av relevante observasjoner knyttet til metode, prosessen, verktøy og ikke minst personell er presentert nedenfor.

Prosjektmetoder i programvareutvikling

"De smidige metodene" var en gjentakende betegnelsene hos flertallet av intervjuobjektene som ble intervjuet. Det smidige metodeverket viste seg å være det mest utbredte, og av de smidige metodene var Scrum det mest fremtredende. En respondent hevdet til og med at konsepter som "Scrum" og "Agile" nærmest var blitt synonymer i dagens programvareutvikling.

Alle de aktuelle selskapene var, i større eller mindre grad, tilknyttet den smidige metodikken. Utbredelsen av disse metodene viste seg også å skape en felles overordnet "smidig-tankegang" hos intervjuobjektene og forankringen av denne tankegangen var svært fremtredende. Selv med et felles tankesett med bakgrunn i smidighet, ble det likevel skapt rom for utforskning og egendefinering, som gjorde at en så variasjon fra bedrift til bedrift. Variasjonen viste seg i form av bruken og egenskapene til metodene.

Intervjuobjekter i topplederstillinger viste derimot at for å kunne skape verdi gjennom det smidige metodeverket ville det kreves et egnet prosjekt. Størrelsen, kompleksiteten og risikoen i et prosjekt måtte eksempelvis tas i betraktning og ville være med på å dempe graden av smidighet. Med bakgrunn i dette var de relevante intervjuobjektene vant til å bruke metoder med elementer fra mer tradisjonelle tankesett, slik som fossefallsmetoden, i symbiose med smidige metoder for å få mest utbytte av prosjektet.

Krav til endringsevne ble stilt til metoden en skulle bruke ettersom intervjuobjektene befant seg i dynamiske miljø hvor ekstern innovasjon eller utvikling ville medføre konsekvenser internt i bedriftene. Med bakgrunn i dette var det viktig at en metodikk hadde evne til å reagere ved å, til en viss grad, selv kunne endre seg under prosjektets livsløp. Muligheten til å ha en fleksibel og dynamisk metodikk var nærmest et suksesskriterie for noen intervjuobjekter. Endringer av metoden kom i form av kontinuerlig utvikling, revisjon og oppgradering, men slike rutiner stilte krav til å ha en bedrift med tilgang på store ressurser.

Som nevnt tidligere kunne ulike faktorer som risiko og størrelse på et prosjekt være avgjørende ved valg av metode. I tilknytning til Scrum og den smidige metodikken var slike metoder oppfattet som "for tynne" ettersom de ikke var omfattende nok eller klarte å skape et godt nok rammeverk rundt prosjektet. Scrum, eksempelvis, kan egne seg godt til å utvikle programvare raskt, men ville ikke kunne klare å dekke "ende-til-ende" i prosjektet. Vi oppfattet også at kunde og deres erfaring med metode ble tatt til betraktning. Kundens grad av involvering i prosjektene gjorde at kundens rutiner rundt metodebruk ville være en faktor som kunne være bestemmende ved metodevalg.

Gjennom intervjuene ble det også klart at det ga fordeler å trekke elementer ut fra metoder for å skape sin egen skreddersydde og sammensatte metode. En slik løsning ville ha mulighet til å svare bedre på virksomhetens krav til metodikk fremfor å følge én metode slavisk. En streng tilnærming som krever at en følger en metode til minste detalj stred også imot den grunnleggende tankegangen til intervjuobjektene, at en må være tilpasningsdyktig. Ved å på denne måten bygge opp metoder med elementer fra ulike steder ble det også vanskelig for noen intervjuobjekter navngi med sikkerhet den fungerende metoden. At den var smidig derimot, var tydelig.

En annen tankegang var at en god metode lignet en katalog med flere prosesser og aktiviteter. For å skape en god metode for sitt prosjekt måtte man plukke ut de elementene som en ønsker fra katalogen, uten å nødvendigvis være bundet til å plukke ut et spesifikt antall. Videre ville det være viktig å kontinuerlig oppdatere og oppgradere listen over elementer for å sikre seg det beste utvalget. Det var derimot klart at ingen metode kan dekke alle behov. En må skreddersy en løsning med bakgrunn i prosjektet for å skape den tilnærmede best egnede metoden.

Mennesker

Dette handler om kompetanse, erfaring og synergi-effekter ved å delta i team i forskjellige roller, typisk i en konsulentfunksjon. Alle våre intervjuobjekter kunne fortelle at de arbeidet i team, men at team-sammensetningen kunne variere basert på type prosjekt, produkt og kunde. Et typisk team som gikk igjen var team bestående av flere utviklere, én eller flere designere, en prosjektleder og en produkteier. I vårt utvalg av respondenter ble vi fortalt at utviklerne stort sett var full-stack utviklere, som betyr at de har erfaring med både back-end og front-end utvikling, samt erfaring med databaser.

Kundedeltakelse er et sentralt element av teamsammensetningen. Graden av samarbeid med kunder er på samme måte som teamsammensetning variabel, og bærer preg av type prosjekt, produkt og kunde. Alle respondenter var samstemte om at et godt samarbeid med kunden er essensielt for å komme fram til et bra, og ikke minst riktig, produkt. "Vi hadde ikke kommet fram til riktig løsning uten dem" sa en av våre respondenter om kundesamarbeid.

Slike engasjement kunne ta forskjellige former. I enkelte prosjekter ble kunde satt inn som prosjektleder eller produkteier, mens i andre prosjekter kunne kunden delta i teamet på en kontinuerlig basis, eller stille opp ved behov for innspill. Det ble også uttrykt frustrasjon ved at kunder til tider kunne være vanskelig å få engasjert, dersom de på forhånd ikke var tildelt en definert rolle i utviklingsprosjektet. Erfaringsmessig var de imidlertid stort sett villige til å stille opp og delta aktivt. Å ha en kunderepresentant i prosjektleder- eller produkteierrollen kan i mange tilfelle være å foretrekke ettersom man da vil ha ressurser som sitter på et internt eierskap og beslutningsmyndighet.

Interaksjon og tverrfaglighet i dag-til-dag arbeidet kan skje i forskjellige varianter. I det daglige samarbeidet ser det ut til at det er kodere og designere som jobber tettest sammen, og at terskelen hos de fleste er meget lav for å ta kontakt og få oppklart ting seg imellom, spesielt der disse jobber samlokalisert. De plassene der designere og kodere ikke jobber samlokalisert ser vi at det kan være tidkrevende å finne frem til rett person, dersom det skulle være spørsmål den ene eller andre veien.

Flere av våre respondenter nevner i forbindelse med tverrfaglighet at måten de jobber på dersom det kreves input fra flere ulike fagdisipliner, eller behov for et større spenn av tverrfaglige innspill og avklaringer, kan minne om Concurrent Design (CCD) konseptet. Her

benyttes workshops med idemyldring og diskusjoner for blant annet å komme opp med hypoteser for å oppnå målene for prosjektet, med ''hjemmearbeid'' mellom planlagte workshops.

Åpenhetskultur, villighet til å dele, disiplin, og et arbeidsmiljø hvor frykten for å feile er fraværende, samt unngå silotenking er suksess-kriterier for å lykkes med å gjennomføre og utvikle seg i tverrkompetanse- prosjekter.

Prosesser

Sentrale elementer i implementering av metodikken er kommunikasjon, møteaktivitet, beslutningsrutiner og sikring av kvalitet.

Kommunikasjon / samhandling

Programvaren ''Slack'' ble av noen presentert som hyppig brukt for samhandling der all skriftlig kommunikasjon skjer, og har nesten tatt over hele den e-post-sfæren. Samtidig blir Jira og Kanbanboards i Jira brukt for oversikt av oppgavene folk jobber med, og som et redskap for å prioritere. Standupmøter hver morgen er ofte brukt, og for mange skjer interaksjon ut over det uformelt på Slack.

Møteaktivitet

Denne delen av prosessen varierte, eksempelvis ble det satt opp møter etter behov, der kun de som hadde direkte nytte av å være med på møtene ble innkalt, eller det ble satt opp faste møtetidspunkt der alle deltakere i prosjektet måtte stille. Det ble gitt uttrykk for at dette i praksis ofte ble en balanse mellom tilgjengelig tid og reelt behov for deltakelse, der mangel på tid og manglende grad av følt behov gjerne var en kilde til frustrasjon. Møter bør imidlertid alltid være godt forberedt, ledet og referert på en strukturert måte.

Beslutninger

Det ble gitt uttrykk for en viss spredning i hvordan beslutnigner ble tatt, men typisk var hierarkiet at prosjektleder (gjerne en representant fra kunde) i samråd med produkteier formaliserte beslutningsprosessen. I såkalte autonome team kunne avgjørelser også blitt tatt basert på en demokratisk prosess, men i hovedsak ville dette gjelde for tekniske spørsmål relatert til selve utviklingen. Forretningsspørgsmål måtte opp i hierarkiet.

Kvalitet

En forutsetning for et godt resultat er at riktig kvalitet blir oppnådd. Det handler blant annet om å ha forstått og kommunisert forventninger slik at de blir innfridd. Digitalisert produkt- og tjenesteutvikling er typisk en iterativ prosess, og erfaringsmessig kan det være kulturforskjeller mellom et utvikler-miljø og bruker-miljøet. Derfor er det viktig at eventuelle gap i forventninger blir adressert og avklart, dette er typisk ivaretatt av en kvalitetsansvarlig hos utvikleren.

Verktøy

Verktøy for samhandling (kommunikasjon, samarbeid og deling) er en viktig del av intervjuobjektenes prosjektgjennomføringsprosess, sammen med fastsatte rutiner for deling, samarbeid, kommunikasjon og koordinering. Disse rutinene skjer daglig og tar i stor grad mye av arbeidstiden til respondentene. Flere slike verktøy for kommunikasjon og samarbeid ble nevnt og er kort beskrevet nedenfor.

Verktøy som Microsoft Teams og Slack har i stor grad tatt over for e-post kommunikasjon. Verktøy som Trello (nettapp for listeproduksjon) eller kanban brett var også populære metoder innenfor prosjektstyring og for å skape oversikt. Dette fant vi også i bruken av kanban metoden gjennom Jira (nettopp for å skape oversikt i programvareutviklingsprosjekter). Innenfor videokonferanser og skjermdelingsverktøy er verktøy som Skype, Microsoft Teams eller Zoom de mest brukte. Kombinasjoner som Microsoft Teams som et helhetlig prosjektverktøy innenfor samarbeid og videokommunikasjon, og Slack som det skriftlige kommunikasjonsverktøyet, var også populære. Egne møterom som legger til rette for skjermdeling og samarbeid i rommet eksisterer til en viss grad, og disse vil være utrustet med projektorer og store skjermer med tilkoblingsmuligheter. Slike rom ble blant annet betegnet som “war-rooms” eller “workshop-rom”.

Verktøy for deling av arbeid og resultater var ulikt avhengig av rollen til intervjuobjektene. Github (verktøy for deling av kode med versjonskontroll) var blant annet et essensielt samhandlingsverktøy for utviklerne. Videre brukte designere verktøy som Abstract (verktøy med versjonskontroll og delingsmuligheter) for å kunne dele design eksternt fra designer til utvikler. Selv om slike verktøy ble brukt i stor grad var muligheten til å sitte i samme kontorlandskap med lav terskel for å kunne spørre eller dele fysisk, viktigere. Enkelheten og

effektiviteten i slike korte møter gjorde det til et essensielt verktøy for kontinuerlig beslutningstaking og samarbeid.

Drøfting

I drøftingen ønsker vi å trekke frem oppgavens tidligere presenterte forskningsspørsmål, diskutere de, og presentere våre svar.

«Hvilke programvareutviklingsmetoder benyttes i dag og hvilke krav stilles til dem?»

Ut fra de gjennomførte intervjuene og den undersøkte teorien har vi skaffet innsikt i dagens situasjon knyttet til normer ved metodevalg samt kravene som stilles til metodene. Intervjuene beskriver et arbeidsmiljø hvor smidige metoder er en fellesnevner for alle prosjekter, både i form av prosessene som utføres eller tankegangen til prosjektdeltakere. Det smidige metodevalget beskrives av intervjuobjektene av å komme fra de underliggende kravene om å legge til rette for kontinuerlig endring gjennom hele prosjektlivssyklusen. I tillegg til den smidige tankegangen ser vi også at mer komplekse prosjekter krever større grad av tradisjonell tilnærming til prosjektstyring. Det var å forstå at denne tilnærmingen kunne eksistere i form av smidige metoder brukt sammen med fossefallsmetoden. Teorien fremstiller også dagens programvareindustri som et miljø hvor flertallet av aktører bruker de smidige metodene (Abrahamsson et al., 2017). Utviklingen mot dagens situasjon kommer som et resultat av nye krav til fleksibilitet og dynamikk som oppstod på starten av 2000-tallet og som har vært aktuelle frem til og med nyere tiår (Dingsøyr et al., 2012). Videre ser en at kombinasjonen av agile og tradisjonelle metodologier i prosjekter eksisterer og er gunstig i de riktige situasjonene (Fernandez & Fernandez, 2008).

Gjennom teorien og de gjennomførte intervjuene ser vi indikasjon til at det i nyere tid er smidige metodeverk som dominerer og i mindre grad tradisjonelle metoder. Det stilles krav om evne til endring hos de smidige metodene samtidig som prosjektets grad av kompleksitet vil være avgjørende i valget om å ta i bruk mer tradisjonelle metoder.

«Hvordan anvendes dagens programvareutviklingsmetoder?»

Intervjuene beskriver en metodeanvendelse preget av å skape skreddersydde løsninger hvor metoden er oppbygd av ulike elementer fra forskjellige metoder. På denne måten kan en sikre

at metoden inneholdt prosesser relevante for aktøren fremfor å ta i bruk en metode hvor flere av momentene ikke hadde evne til å skape forretningsmessig verdi. Videre ble det beskrevet at metodene måtte kunne kontinuerlig endres og på den måte kunne tilby det beste og mest aktuelle utvalget av egenskaper. "Agile tailoring" er et begrep som brukes i teorien for å beskrive en slik tankegang (Conboy & Fitzgerald, 2007). Det eksisterer også i teorien en oppfatning om at tilpasning, fragmentering og deretter utskiftning av utviklingsmetoder innenfor det agile metodeverket er kritisk i forhold til vellykketheten av prosjektene (Conboy & Fitzgerald, 2007). En antagelse om at et smidig metodeverk skal være formbart sies også av noen å være en selvfølge (Conboy & Fitzgerald, 2007).

Vi ser gjennom teorien og intervjuene at skreddersydde løsninger kan se ut til å være den fremtredende anvendelsen av metoder. Ved å bruke slike løsninger ønsker aktørene å legge til rette for selskapets egne krav til metode samtidig som en kan øke sannsynligheten for vellykkede prosjekter som beskrevet av teorien.

«I hvilken grad samsvarer dagens praksis med prinsippene anvendt i CCD?»

Målet her var å kartlegge i hvilken grad dagens praksis samsvarte med prinsippene for CCD basert på input fra våre respondenter og teori på feltet, samt se på forskjeller i forhold til prosjekter der CCD mer typisk blir anvendt.

CCD prosjekter er ofte karakterisert som ingeniørprosjekter med høy grad av kompleksitet og involverer forskjellige kompetanseområder og fagdisipliner. I Norge er CCD blitt brukt i oljeindustrien og Dette er prosjekter med investeringer i milliardklassen som går over flere år, har leveranser med stor geografisk spredning og ofte med betydelig innhold av ny teknologi. Denne graden av kompleksitet innebærer et behov for felles forståelse på tvers av kompetanseområder og sanntids informasjonsutveksling for best å kunne holde tritt med stramme tidsplaner, kostnadsmål og kvalitet. Innen offshore har det lenge vært en praksis med multidisiplinære aktiviteter slik som HAZID (Hazard IDentification), HAZOP (Hazard and OPerability) studier, og FMECA (Failure Mode, Effect and Criticality Analysis) for å sikre en omforent forståelse og treffe gode beslutninger. Her er kompetanse fra en rekke fagfelt som geologi, hydrodynamikk, styrkeberegninger, elektro og instrumentering, prosesstyring

involvert, for å nevne noen. Følgelig er det en klar gevinst ved å kjøre denne type prosjekter med relativt stramme tøyler for å kunne ha nødvendig kontroll på gjennomføringen.

Når dette regimet skal sammenliknes med krav til prosjektgjennomføringsmetoder og praksis for utvikling av programvare, ser vi tegn til at skillet mellom dem er at programvareutvikling typisk vil ha et noe mindre tverrfaglig arbeidsomfang enn de mer klassiske ingeniørprosjektene som CCD opprinnelig er utviklet for.

Utvikling av programvare kan riktignok også være tverrfaglig, men i et mindre omfang og med fokus på koding og andre elementer som at relevante bestemmelser i spesifikasjoner er ivaretatt til at brukergrensesnitt er funksjonelle.

Det kan tyde på at dagens praksis har en del overlapp med prinsippene anvendt i CCD. Elementer av tverrfaglig utveksling ble adressert gjennom workshops ved behov, men ikke i samme omfang og format som for CCD, og mindre omfang reduserte behovet for formell sanntids samarbeid.

«Hva kan CCD tilføre av verdi inn i dagens praksis?»

Til tross for det overnevnte standpunktet ser vi for oss at CCD kan benyttes til god effekt i faser av programvareutvikling der tverrfaglighet er mer fremtredende eller i større prosjekter som viser seg å ha en uvanlig stor grad av kompleksitet der mange aktører fra ulike fagfelt er involvert. Dette er i bakgrunn at en kan se på CCD som en mer rigid metode. Egenskaper som fastsatte møterutiner med lengre tidsintervall mellom sesjoner, strenge krav til oppmøte på sesjonene og vektlegging av tverrfaglig samarbeid i sanntid, støtter opp under dette.

Med dette tatt til betraktning kan det vise seg verdifullt å se på CCD i rollen som den tradisjonelle metoden i et samarbeid med et smidig metodeverk. Denne tankegangen så vi både eksisterte gjennom intervjuene, samt i litteraturen gjennom den hybride løsningen som beskrevet av Fernandez og Fernandez (Fernandez & Fernandez, 2008). En antagelse blir at prosjektet oppfyller karakteristikkene hos egnede prosjekter innenfor kompleksitet/størrelse, grad av risiko og linearitet, og at CCD er godt nok egnet i rollen som en tradisjonell metode (Špundak, 2014). Resultatet kan bli at CCD vil fungere som et rammeverk rundt prosjektene ved

å skape en tyngde i fastlåste prosesser og videre vil prosessene innenfor rammeverket utføres gjennom det smidige tankesettet.

Avslutning

Gjennom denne oppgaven har vi adressert i hvilken grad Concurrent Design (CCD) har en naturlig plass i utviklingen av programvare. Det er gjort ved å skaffe informasjon om dagens praksis for utvikling av programvare, og hvordan den sammenfaller med gjennomføring av typiske CCD prosjekter. Videre har vi sett på om og hvordan CCD, eller elementer av denne metodikken, kan bidra til å håndtere utfordringer eller gagne dagens praksis for utviklingen av programvare.

Fra vår forskning er det blitt tydelig at såkalt smidige utviklingsmetoder i stor grad foretrekkes, og er "de facto" fremgangsmåten for utviklingen av programvare i dag. Dette skyldes til dels at metodene responderer godt til krav om endring, at utvikling av programvare er 'vektløs', i den forstand at endringer og optimalisering kan gjøres uten betydelig ressursbruk, og at spesialistene (utviklerne) som utfører selve utviklingsprosessen har kunnskap og frihet til å velge og anvende relevante elementer fra ulike metoder. Dette skjer normalt i samråd med designer og kunden for best å sikre at forventningene blir oppnådd, og denne prosessen kan karakteriseres som relativt fri og fleksibel.

Det er en del fellesnevnere mellom arbeidsprosessene for CCD og utviklingen av programvare, som for eksempel workshops og idemyldring. CCD anvender imidlertid en mer rigid og systematisk struktur, karakterisert ved etablerte prosesser, multidisiplinære aktiviteter, integrerte design modeller, fasiliteter og infrastruktur med etablerte roller for sanntids interaksjon, slik det er beskrevet i de åtte prinsippene for CCD. Et CCD team består normalt av en fasilitator, sekretær, domene eksperter, prosjektleder og kunderepresentant(er) som er dedikert til en felles framgangsmåte.

CCD kan derfor bli for rigorøs til direkte å anvendes i programvareutviklingsprosjekter sett opp imot dagens praksis slik vi ser det, men en form for integrering av programvareutviklingsteam i CCD prosjekter kan være på sin plass for best å sikre en felles forståelse av programvarebehov som en del-leveranse i et større prosjekt.

Med bakgrunn i funnene ser vi muligheten for videre forskning innenfor anvendelse av CCD som metode i programvareutviklingen. Vi mener det kan være interessant å se nærmere på CCD i rollen som den mer tradisjonelle metoden i en tradisjonell-smidig hybrid metode.

Ettersom vår oppgave har tatt for seg treenigheten mellom mennesker, prosesser og verktøy vil også videre forskning på egnetheten av de 8 prinsippene fra oppbygningen av CCD kunne avdekke tydeligere egnetheten til CCD i denne typen utvikling.

Kritikk

Ved å se tilbake på resultat og gjennomføring av oppgaven finner vi noen punkter vi kan komme med kritikk. Som følge av mangel på innsikt i dagens programvareutviklings-situasjon ser vi at utforming av intervjuguide og valg av intervjuobjekter ikke nødvendigvis er optimal for å svare på problemstillingen. Intervjuobjekter i andre roller i bedriftene kunne sitte på nyttig innsikt som vi ikke hadde tilgang til, samt at vår intervjuguide ikke har mulighet til å avdekke alle områder av interesse. Det siste påvirkes også av vår smale erfaring innenfor utførelse av intervjuer, der vi kanskje burde brukt litt tid på test-intervjuer i forkant av intervjuene våre. Likevel har vi klart å få tak i et utvalg objekter spredt utover hele hierarkiet i et selskap, noe som vi ser på en styrke i oppgaven. Videre er respondentene ansatte hos konsulentbedrifter og hvis oppgaven hadde inkludert intervjuobjekter fra «kunde»-siden og selskaper med in-house-utviklere kunne oppdaget andre synspunkter. Til slutt kan vi nevne at oppgaven har måtte utformet seg gjennom prosjektlivsløpet. Dette er på bakgrunn av, som nevnt tidligere, mangel på innsikt i industrien vi undersøker. En mer tydelig oppgave fra starten av prosjektet kunne ha gitt andre synspunkter eller trukket frem nye argumenter.

Kilder

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2017). Agile software development methods: Review and analysis. *arXiv preprint arXiv:1709.08439*.

Banfield, R., Lombardo, C. T., & Wax, T. (2015). *Design sprint: A practical guidebook for building great digital products*: " O'Reilly Media, Inc.".

Bass, J. M. (2016). Artefacts and agile method tailoring in large-scale offshore software development programmes. *Information and Software Technology, 75*, 1-16.

Campanelli, A. S., & Parreiras, F. S. (2015). Agile methods tailoring–A systematic literature review. *Journal of Systems and Software, 110*, 85-100.

Cockburn, A. (2006). *Agile software development: the cooperative game*: Pearson Education.

Conboy, K., & Fitzgerald, B. (2007). *The views of experts on the current state of agile method tailoring.* Paper presented at the IFIP International Working Conference on Organizational Dynamics of Technology-Based Innovation.

Cunha, M. P. E., & Gomes, J. F. (2003). Order and disorder in product innovation models. *Creativity and innovation management, 12*(3), 174-187.

Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. In: Elsevier.

Fernandez, D. J., & Fernandez, J. D. (2008). Agile project management—agilism versus traditional approaches. *Journal of Computer Information Systems, 49*(2), 10-17.

Highsmith, J. A., & Highsmith, J. (2002). *Agile software development ecosystems*: Addison-Wesley Professional.

ISO. (2018). THE PROCESS APPROACH IN ISO 9001:2015.

Jørgensen, M. (2018). *Do agile methods work for large software projects?* Paper presented at the International Conference on Agile Software Development.

Nambisan, S., & Wilemon, D. (2000). Software development and new product development: Potentials for cross-domain knowledge sharing. *IEEE Transactions on Engineering Management, 47*(2), 211-220.

Oxnevad, K. I. (2000). The NPDT–The next generation concurrent design approach. *Proceedings of EuSEC, 2000*.

Permana, P. A. G. (2015). Scrum method implementation in a software development project management. *International Journal of Advanced Computer Science and Applications, 6*(9), 198-204.

Seale, C. (1999). Quality in Qualitative Research. *Qualitative Inquiry, 5*(4), 465-478. doi:10.1177/107780049900500402

SIMTANO. (2020a, 2020). CONCURRENT DESIGN – THE METHOD.

SIMTANO. (2020b). THE SIMTANO TEAM.

Smith, P. G., & Reinertsen, D. G. (1992). Shortening the product development cycle. *Research-Technology Management, 35*(3), 44-49.

Sohlenius, G. (1992). Concurrent engineering. *CIRP annals, 41*(2), 645-655.

Špundak, M. (2014). Mixed agile/traditional project management methodology–reality or illusion? *Procedia-Social and Behavioral Sciences, 119*(1), 939-948.

Stober, T., & Hansmann, U. (2010). *Best Practices for Large Software Development Projects*: Springer.

Takeuchi, H., & Nonaka, I. (1986). The new new product development game. *Harvard business review, 64*(1), 137-146.

Tjora, A. (2012). *Kvalitative forskningsmetoder i praksis. 2. utgave*.

Vilkki, K. (2010). *When agile is not enough.* Paper presented at the International Conference on Lean Enterprise Software and Systems.

Winner, R. I., Pennell, J. P., Bertrand, H. E., & Slusarczuk, M. M. (1988). *The role of concurrent engineering in weapons system acquisition*. Retrieved from

Woodward, E., Surdek, S., & Ganis, M. (2010). *A practical guide to distributed Scrum*: Pearson Education India.

Vedlegg

Intervjuguide

Fase 1: Rammesetting

Løst prat (5 min)

- Uformell prat
- Informer om ev. opptak, sørg for samtykke til ev. opptak

Informasjon (5-10 min)

- Oppbygningen av intervjuet
- Si litt om temaet for samtalen (bakgrunn, formål)
- Forklar hva intervjuet skal brukes til og forklar taushetsplikt og anonymitet
- Spør om noe er uklart og om respondenten har noen spørsmål
- Ønsker du å opptre anonymt eller kan vi bruke informasjon som identifiserer deg?
- Presentere Concurrent Design i korte trekk
- Tillatelse til taleopptak?

--Start opptak--

Fase 2: Erfaringer

Overgangsspørsmål: (15 min)

- Hvor jobber du, og hva jobber du med?
- Kort om bakgrunn

Fase 3: Fokusering

Nøkkelspørsmål: (50-60 min)

1. Erfaring rundt produksjon av digitale produkter
    a. Hva slags produkter/prosjekter har du arbeidet med tidligere?

2. Hvilke prosjektmetoder benytter dere?
    a. Hvorfor disse metodene? Styrker/svakheter? Negative/Positive erfaringer?
    b. Hvilke egenskaper har disse metodene som er viktige for dere?
    c. Hva er bra med metodene/hva har fungert bra i praksis, utfordringer av metodene?
    d. Bruker dere andre metoder for mer komplekse systemer/produkter?
3. Hvordan jobber dere i team?

4. Beskriv møtevirksomheten deres

    a. Hvem sitter i møtene?
    b. Hvor ofte krever et prosjekt at en gjennomfører samlinger/møter?
    c. Hva er produktene av møter dere har?
    d. Beslutningstakning i møter, er det myndighet til å gjennomføre beslutninger?

5. Hvilke faste prosesser/rutiner har dere i prosjekter? (Typ oppsummeringsmøter, planleggingsmøter)

6. Tar dere i bruk verktøy utviklet for samhandling? Hvilke?

7. Jobber dere tverrfaglig?
    a. Hvordan legges det til rette for tverrfaglig samhandling?
    b. Legges det til rette for tverrfaglig samarbeid i form av fasiliteter, faste møter, delingsverktøy/andre verktøy?
    c. Får dere uttelling for det tverrfaglige samarbeidet?

8. Hvor delaktig er kunden i prosessen? På hvilken måte deltar kunden?

9. Syn på suksessfaktorer til digital produktutvikling
    a. Teamarbeid
        i. Hvordan oppnå godt teamarbeid?
    b. Felles visjon
        i. Hvordan oppnå felles visjon?
    c. Effektivitet i samhandling
        i. Hvordan oppnå effektivitet?

10. Hva er bra med metodene/hva har fungert bra i praksis, utfordringer av metodene
    a. Positive og negative egenskaper

Fase 4: Tilbakeblikk

1. Oppsummering (ca. 15 min)

- Oppsummere funn
- Har jeg forstått deg riktig?
- Er det noe du vil legge til?

NTNU

Kunnskap for en bedre verden