Forelesning 3: Prosesser (forts.), Prosjektledelse

Evolusjonære utviklingsprosesser

Introduksjon

Det finnes flere evolusjonær utviklingsprosesser. Alle er basert på samme overordnede prinsipp men avviker på andre områder. Prinsippene er blitt praktisert minst like lenge som fossefallsmodellen, men det var først på midten av 80-tallet at denne praksisen ble nedfelt i spesifikke navngitte instanser (disse blir normalt omtalt som metoder). På dette tidspunktet hadde allerede fossefallsmodellen fått et globalt fotfeste. I Norge var fossefall mye anvendt til langt ut på 90-tallet. Det er først nå i 2009 at bransjen i Norge snakker om det store Scrum-året. Som nevnt, finnes det flere ulike navngitte prosesser (metoder) som vi skal se på. Dette er prosesser som gjerne har en bestemt opphavsmann, og prosessene er gjerne godt og detaljert beskrevet i egne bøker. Innenfor noen av prosessene tilbys det en rekke kurs, foredrag, og ikke minst sertifiseringer (derav betegnelsen Scrum-året).

Innenfor evolusjonære prosesser er det nyttig å vite at noen prosesser kun beskriver gjennomføring uten å beskrive hvordan konstruksjonsarbeidet skal foregå. Med gjennomføring menes her hvordan arbeidet skal ledes men ikke noe om for eksempel hvordan programmererne skal jobbe for å løse sine oppgaver. Prosesser som kun beskriver hvordan arbeidet skal ledes må kompletteres med nødvendige metoder for nær sagt alle de sentrale fasene fra kravinnsamling og kravanalyse til test.

Gjennomføring

En enkel måte å tenke på gjennomføringen av en evolusjonær prosess er å dele et prosjekt inn i flere sekvensielle mini-prosjekter eller iterasjoner. Hver iterasjon har en varighet på en uke til rundt fire uker. De fleste velger en fast lengde for alle iterasjonene. Hver iterasjon bør ha et mål og det lages en detaljert plan i i forkant eller i starten av iterasjonen. Planleggingen foregår gjerne ved at alle kjente oppgaver prioriteres og prosjektgruppen går i gang med de høyest prioriterte oppgavene. Hvis vi f eks har en prosjektgruppe bestående av 5 personer(prosjektleder og utviklere), vil disse fra starten av prosjektet begynne med første iterasjon og gjennomføre kravinnsamling og kravanalyse, design, programmering, og test. For enkelthets skyld kan vi tenke oss at de gjøres i denne rekkefølgen. Alle sammen skal utføres i løpet av de 4 ukene iterasjonen varer. Når første iterasjon er avsluttet begynner prosjektgruppen på andre iterasjon. Slik fortsetter det til antall planlagte iterasjoner er gjennomført. Merk at fremstillingen av mini-prosjektene er sterkt idealisert. I moderne prosesser finner en normalt ikke spor av en slik streng inndeling i hver iterasjon. Det betyr at noen i prosjektet gjerne kan jobbe med design mens andre programmerer. Det er imidlertid slik at hver oppgave prosjektet skal løse bør gjennomgå fasene fra kravanalyse til test. Aktiviteter eller større oppgaver blir brutt ned til flere mindre oppgaver av forholdsvis kort varighet slik at gjennomføring er enkel å styre. Det betyr at viktige deler av systemet ikke nødvendigvis blir ferdigstilt i løpet av en iterasjon. Iterasjonen vil i så fall bidra til at en eller noen få deler av den større oppgaven løses.

Generelt har forskning vist at evolusjonære prosesser er bedre egnet enn fossefallsmodellen. Det kan derfor synes noe merkelig å snakke om ulemper og fordeler kontra andre prosesser. Det er imidlertid

noen utfordringer knyttet til evolusjonære prosesser som det er nyttig å kjenne til. Merk at disse ikke nødvendigvis er godt løst i fossefall heller. Et viktig aspekt ved evolusjonære prosesser er at de flytter en del av ansvaret fra prosjektleder til prosjektgruppen. Bakgrunnen for dette er at man antar at de som skal utføres oppgavene har et bedre utgangspunkt for å fordele arbeidet på en fornuftig måte. På den måten får hver deltager mer ansvar og blir forhåpentligvis mer dedikert for å løse oppgaven han eller hun har påtatt seg. Det krever mer av prosjektgruppen og en prosjektgruppe som ikke klarer å organisere seg selv vil fungere mindre bra enn hvis den ble styrt av en sterk leder. En annen svakhet ved evolusjonære prosesser er at de tenderer til å gi lite støtte til arbeidet med arkitektur og vedlikeholdbarhet. Dette kan føre til at disse aktivitetene får for lite oppmerksomhet. De fleste evolusjonære prosesser fokuserer mye på funksjonalitet. Det er ikke overraskende fordi de nettopp er laget for å sikre at vi lager det rette systemet. At vi konstruerer systemet på den riktige måten blir dermed sekundært. I dette ligger det også at ikke alle smidige metoder har som mål å beskrive hvordan dette skal løses. Dette er imidlertid ikke alle klar over og på den måten kan man risikere at arbeid med f eks arkitektur (dvs strukturen i systemets tekniske oppbygning) ikke blir gjennomført på en god måte.

Instanser av evolusjonære prosesser

Utviklingen av evolusjonære prosesser har pågått over lang tid. Ulike instanser har blitt utviklet av uavhengige opphavsmenn. Til tross for at de overlapper på overordnede prinsipper så har de forholdsvis ulike fokus. RUP er f eks svært konstruksjonsdrevet og vektlegger bruk av objektorientert analyse og design. Dette skal understøttes av flere avanserte verktøy. Underforstått forventes det at forholdsvis mye tid skal brukes til å arbeide med å utvikle objektorienterte diagrammer i f eks UML. Resultatet er en rekke objektorienterte modeller som beskriver både overordnet og detaljert arkitektur, og som brukes som utgangspunkt (blåkopier) for programmeringen. Sammenlikner vi med fossefall kan en se en tendens til fortsatt å fokusere på produksjon av dokumentliknende leveranser. I såkalte agile (smidige) metoder er det lagt betydelig mer vekt på å bruke tiden på programmering og test. Det legges mindre vekt på forarbeidet og planleggingen. Tanken bak dette er at vi vil allikevel ikke vil klare å komme frem til en tilstrekkelig presis arbeidsskisse for hva som skal utvikles. Desto raskere vi kan få frem skjermbilder (prototyper), desto raskere kan vi finne ut hva vi faktisk har behov for og hvordan det skal presenteres for sluttbruker. Hvis vi ser på mantraet for smidige metoder, ser vi at oppmerksomheten primært er rettet mot en direkte interaksjon mellom prosjektmedlemmene. I motsetning til fossefall, fokuserer smidige metoder på at partene sitter sammen og diskuterer i samme rom. I fossefall vil gjerne en part utarbeide et detaljert og stort dokument som så sendes til en godkjenning hos andre parten (som gjerne er kunden).

Scrum og XP

Scrum og XP er kanskje de modellene som er mest benyttet i dag. Scrum er en modell som fokuserer på gjennomføringen, hvilke møter som skal arrangeres, hvor lenge hver iterasjon (kalles sprint i Scrum) varer, hvilke roller som skal brukes, hvordan planleggingen skal foregå. XP er en mer ingeniørrettet modell som også gir en mer utførlig beskrivelse av programmeringsteknikker som f eks par-programmering der to programmerere sitter sammen på en PC. XP beskriver også noe om hvordan en skal gå frem for å sikre at arkitekturen i systemet videreutvikles og hvordan dette skal testes. Scrum kan på den måten, mer betraktes som et rammeverk der prosjektet selv må finne egnede metoder som f eks de som er foreslått i XP. Derfor er kombinasjonen av Scrum og XP mye

brukt. Da benyttes gjennomføringsmodellen i Scrum mens de ulike teknikkene som par-programmering osv. er hentet fra XP.

Prosjektarbeid

Prosjektledelse og prosjektarbeid er et klassisk fag med forholdsvis lange tradisjoner innen andre ingeniørdisipliner. Mye av det som gjøres innen prosjektarbeid i systemutvikling kommer derfra. Hovedmålet er å kunne planlegge hva som skal gjøres og når, hvem som skal gjøre hva, samt avklare hvordan fremdrift skal måles. Det er rimeligvis mange andre aspekter ved prosjektledelse men denne delen av kurset omhandler kun de mest elementære delene. Forelesningene om kontrakter går imidlertid vesentlig dypere på noen aspekter.

I de smidige modellene er den klassiske tilnærmingen til prosjektledelse delvis forlatt og erstattet av en noe enklere modell for prosjektstyring. I klassisk prosjektledelse har f eks prosjektleder en sentral rolle med mye myndighet. I Scrum er denne rollen fjernet og erstattet av en annen rolle (Scrum master) som fungerer som en fasilitator og ikke som en leder. Dette henger sammen med at den utførende delen av prosjektgruppen (dvs programmerere, testere, etc) selv må ta en del avgjørelser. Dette er avgjørelser som en klassisk prosjektleder normalt har ansvaret for. De smidige modellene har nærmest forsøkt å kutte bort så mye av den klassiske prosjektstyringen som mulig, fordi dette er arbeid som anses som lite nyttig og dermed vurdert som overflødig. Målsettingen er å gjennomføre prosjekt-arbeidet på en så enkel måte som mulig, men samtidig beholde de sentrale elementene. Eksempelvis er nettverksdiagrammer ikke anbefalt brukt i smidige metoder. I Scrum er det f eks ikke viktig å holde kontroll over hvor mange timer som er brukt, derimot er det viktig å hele tiden estimere omfanget av *gjenstående* arbeid. Når en gitt oppgave startes og når den avsluttes er heller ikke så sentralt. Det viktigste er at prosjektgruppen til enhver tid arbeider med de høyest prioriterte oppgavene. Under forutsetning om at systemutviklingen og prosjektgruppen fungerer optimalt vil de da gjøre det riktige til enhver tid. Hvorvidt dette er en nyttig tilnærming kommer vi ikke inn på her. Det er imidlertid viktig å vite at ulike utviklingsmodeller har forskjellige tilnærminger til prosjektarbeidet. I denne forelesningen kommer vi i liten grad innpå dette og formålet er mer å formidle hva som er felles, samtidig som deler av klassisk prosjektarbeid er inkludert.

En annen viktig forskjell mellom klassisk og smidig tilnærming er at i en klassisk tilnærming planlegger man normalt over en mye lenger periode. Parallellen til fossefall er tydelig. Alle planer skal utarbeides nærmest før arbeidet starter. I smidige modeller planlegger man normalt en iterasjon om gangen. Her dukker det da opp en konflikt fordi man normalt har et begrenset budsjett og er tvunget til å avslutte og levere et godt system etter en viss tid. Tanken bak smidige metoder er at hvis vi til enhver tid jobber med de viktigste oppgavene og prosjektgruppen for øvrig fungerer godt vil vi få mest mulig igjen for investeringen. Hvis vi derimot følger en klassisk modell med fossefall vil sjansen for å mislykkes øke og en for stor del av budsjettet benyttes til å produsere dokumenter som vi ikke behøver. Men et vesentlig problem i smidig tilnærming, er at manglende detaljbeskrivelser av sluttresultatet i et prosjekt ofte skaper vanskeligheter i å utforme en kontrakt som kunden kan akseptere. Hvordan dette kan løses vil bli beskrevet senere i dette kurset.