HØGSKOLEN I AGDER

Kurskode/Fag : IS-402
Kandidatnr. : 3403
Dato : 28/11-07
Ark nr. : 1 av 10

Denne kolonne er forbeholdt sensor.

Oppgave 1

Systemutvikling er vanskelig bl.a. fordi man ikke utvikler noe konkret, men programvare i en eller annen fasong.
Utvikler man på andre områder, f. eks innen bygg, kan man se bygningens vekst fra dag til dag, det er lettere å være intuitiv i forhold til å oppdage ting som må endres, og det er lettere å forstå konsekvensomfanget av feil man har gjort og rettelser som må gjøres. For en kunde er det også lettere å se at hvis grunnmuren ble feil konstruert, og selve bygget er påbegynt, så vil det å skulle rette grunnmuren ha store konsekvenser for resten av bygget, for tiden man bruker, og for kostnadene ved feilopprettingen eller endringen.
I en applikasjon er ikke slike konsekvenser like lett å få øye på. Endringer som kan virke små og overflatiske, f. eks. innholdet/hendelsen ved et klikk på en knapp, kan kanskje i virkeligheten bety en endring i selve den logiske oppbyggingen av applikasjonen, ja sågar en endring av teknologien som ligger til grunn. Da er det ikke lett å få aksept for betydelige kostnadsendringer, dvs økninger, når konseptet er så vanskelig å forstå.
Det koster å få et perfekt system, man kan redusere prisen, men dermed må man også redusere funksjonaliteten.

Denne kolonne er forbeholdt sensor.

Noen vanskeligheter er knyttet til selve softwareutviklingens natur, slik som kompleksitet, konformitet, endringer (særlig endringen som gjøres etter at systemet er utviklet), og det at software i seg selv er usynlig og ikke lett lar seg visualisere. (Brooks)
Selv er jeg opptatt av dette med visualiseringen. Jeg tror at om man skulle greie en dag og oppdage en "Silver Bullet" for systemutviklingsmetoder, så måtte det være om man klarte å utvikle en liksom-visualisering, som ikke nødvendigvis måtte være perfekt, men som ville gi kunden et "bilde" av "systemutviklingsbygningen", slik at konsekvenser av endringer ble mer intuitivt, slik som med ekte bygninger.

Andre vanskeligheter er mer tilfeldige eller i hvert fall ventet.
Programmeringsspråkene er f.eks kommet så langt at man ikke lengen har den enorme kompleksiteten man hadde i starten. Nå er det lettere å forstå selve språket, og dermed lettere å programmere riktig.
Det er også blitt bedre samsvar applikasjonene i mellom, de kan bruke samme bibliotek osv., standardene er mer utbredt.

Denne kolonne er forbeholdt sensor.

Oppg 1 forts.

Prosjekter innen systemutvikling mislykkes i en grad som ikke er toleverbart innenfor andre typer prosjekter. Det er ikke unormalt at systemer leveres alt for sent, til en høyere kostnad enn først antatt, og med en dårligere funksjonalitet enn forutsatt.
Innenfor systemutvikling er det et stort problem at man ikke kan lage en endelig kravspesifikasjon i utgangspunktet, i hvert fall ikke alltid, og det skjer stadig endringer underveis, med uoversiktlige konsekvenser.
I tillegg tar systemutvikling forholdsvis lang tid, (noe Brooks ikke er enig i, han mener det er hardware-teknologien som går uforholdsmessig fort, og det kan han jo ha rett i, men ikke desto mindre er resultatet det samme.) noe som gjør at de kravene kundene hadde i starten av prosjektet ikke nødvendigvis er de behovene de har mot slutten av prosjektet.
Prosjektmedlemmene har kanskje ikke den rette kompetansen, de sitter muligens for langt fra hverandre, kommunikasjonen går ikke som den skal, informasjon kommer ikke fram til rett person, man har ikke evaluert forrige prosjekt med henblikk på å lære av feil, man har ingen felles forståelse av målet ei heller veien dit, noen prosjektmedlemmer kan ha helt andre agenda enn man tror, man har ikke støtte fra ledelsen, man har for mange andre oppgaver i tillegg, budsjettet er ikke

Denne kolonne er forbeholdt sensor.

oppg 1 forts

stort nok, noen kan sabotere med vilje, det er en uendelig liste med muligheter for å feile, allikevel går mange prosjekter bra heldigvis!

En metode kan hjelpe prosjektet til å nå målet. En metode har retningslinjer for hvordan ting skal gjøres, i hvilken rekkefølge, sier noe om hvem som har ansvar for hva, kanskje hvilke teknikker som skal brukes, avklaring av begreper for å gi prosjektmedlemmene en felles forståelse av terminologien som brukes, man unngår dermed misforståelser av den typen, og har en fremdriftsplan som gjør at man forhåpentligvis får med seg det neste av de aktivitene man må, innenfor hvert steg. Man kan dermed lettere unngå å glemme noe.

Denne kolonne er forbeholdt sensor.

Oppg 2

Til å begynne med var programmererne enevådene på systemutviklings område. Programmeringsspråkene var svært komplekse og vanskelige å lære, de ble derfor forbeholdt eliten. Og det var lang avstand mellom dem og de senere brukerne. I lang tid var programmereren også brukeren, og den første anstrengelsen kom da man skulle utvikle for andre. Man fikk en kommunikasjonsutfordring. Det fantes ingen formell utdanning for utviklere og det fantes heller ingen programmeringsstandarder, noen utviklere lå til og med sjela si i å gjøre det så komplekst som mulig fordi det fyldet dem at ingen andre kunne lese koden.

Da andre fagområder oppdaget hva systemer kunne gjøre for dem, kom plutselig kravene til systemet andre steder fra, istedet for fra programmerne selv.

Det fantes ikke estimeringsmetoder for kostnader knyttet til utvikling, og programmerne måtte derfor godta det kunden mente det ville koste.

Siden informasjonsflyten mellom programmerer og bruker var dårlig, og man heller ikke hadde noen god kostnads estimat metode, ble det fort utfordringer i forhold til systemer som ikke ble levert i tide, det kostet for mye, og funksjonaliteten var dårligere enn forventet.

Denne kolonne er forbeholdt sensor.

Disse problemene førte til at man fikk fokus på analyse og design i forhold til systemutvikling, og ikke bare programmeringen. Som utgangspunkt brukte man teknikkene som fram til da hadde vært brukt innenfor ingeniør-fagene.
Man videreutviklet etterhvert teknikkene, ved først og tildels skille programmeringen helt fra analyse- og designdelen, så ved å ta i bruk selve datamaskinen som en del av analyse-verktøyet.
Dette foregikk svært tidlig i utviklingsperioden, og ettersom tiden gikk samlet man seg om en livssyklus metode, forøvrig nokså nær opptil ingeniør-faget også den.
Metoden tok for seg planlegging, analyse, design og implementering.
Poenget var å avslutte en fase før den neste begynte, den såkalte fosse fallsmetoden.
Man hadde kommet til den delen som i dag går under fellesbenevnelsen "strukturerte metoder"
Felles for disse metodene er at de nettopp benytter fosse falls prinsippet, man må gjøre en fase helt ferdig, før man starter på neste.
I strukturerte metoden benytter man seg av top-down - prinsippet, dvs at man bryter systemet ned i mindre moduler, hvor hver modul gjør en ting, modulene er uavhengig av hverandre, og innmaten i modulene er skjult. På denne måten kan man forenkle planleggingen og flere kan jobbe på systemet

Denne kolonne er forbeholdt sensor.

samtidig.
I analysedelen benytter man seg av ulike teknikker slik som dataflytdiagrammer, datamodelldiagrammer osv
Slike metoder er gode for systemer der krav-spesifikasjonen ikke endres underveis, der man er avhengig av høy kvalitetssikring og der man har veldig behov for kontroll. Det kan f.eks være i flysystemer, systemer som overvåker kritisk syke mennesker, militære overvåkingssystemer osv
Ulempene er at man ikke har noen gode teknikker for å lage designet basert på analysen, metodene har aldri vært skikkelig testet og innvendingen til Storer:
Du skal bryte ned en abstrakt ukjent størrelse, til flere små, som skal være mer detaljerte og kjente. Men hvordan kan du det når den originale størrelsen er ukjent?
Han sier svaret er intuisjon, erfaring, prøving og feiling.

Det syns jeg er et paradoks, fordi i strukturerte metoder tilstreber man nettopp å gjøre "oppskriften" så detaljert at "hvem-som-helst", skal kunne utvikle ved å følge den. Ingen plass for prøving og feiling med andre ord.
Erfaringsmessig har han allikevel rett.

Denne kolonne er forbeholdt sensor.

De agile metodene kom som et svar på den rigide rammen som ble satt opp rundt de strukturerte, pluss de stadige endringen i kravspesifikasjon man opplevde og som de strukturerte metodene ikke kunne takle. Agile metoder kan i noen tilfelle beskrives helt ned til minste detalj, f. eks hvordan man sitter, alene, to eller flere, nøyaktig hvor lenge møten skal vare osv.
Kjernen i det hele er at man skal takle endringen som kommer underveis. Programkoden sees på som den viktigste delen av dokumentasjonen.
Her også isolerer man ett og ett krav, eller modul om man vil, men man løser det ene funksjonskravet helt og holdent, senere det til kunden - som forøvrig har vært med underveis - og lar kunden prioritere neste viktige funksjon.
På denne måten kan kunden ombestemme seg underveis, han kan justere egne behov og krav, og hele tiden vite status på prosjektet. Han vet også mer om hvor lang tid ting tar, hvor mye det vil koste osv, og det gjør at han selv kan stoppe prosjektet når han ser at gjenstående funksjoner i liten eller ingen grad vil hjelp bvsinessen hans.
Ved en slik deprofesjonalisering av systemutvikling flyttes fokus over på menneskene.

Denne kolonne er forbeholdt sensor.

Oppg 3

Figuren viser sammenhengen mellom kvalitet, omfang, kostnad og tid. Endrer du en av de tre siste og holder de andre konstant, så vil kvaliteten endres.

I Agile metoden er dette forsøkt taklet ved at man holder kostnad og tid konstant, og man mener at det da er mulig med en kontrollert variasjon i funksjonalitet, dvs kvalitet.
Man holder tiden på en iterasjon hellig, prisen er også satt, og så prøver man å utvikle nøyaktig den funksjonaliteten man skal innenfor den ene iterasjonen. Når tiden er ute stopper man utviklingen. Det som er ferdig kan leveres, det som mangler på funksjonaliteten inngår nå som en del av det som gjenstår, og skal prioriteres på nytt på linje med andre gjenstående funksjonaliteter.
For kunden er dette bra fordi han til enhver tid har god oversikt over ferdigstillte og gjenstående behov, samtidig som det gir ham mulighet til å prioritere på nytt og på nytt i takt med prosjektets status og kundens egne behov.
Den tiden man har sagt man skal bruke på en iterasjon kan være testet ut på forhånd ved at man kjører en test-iterasjon for å få en følelse av prosjektets tyngde og størrelse

HØGSKOLEN I AGDER

Kurskode/Fag : IS-402
Kandidatnr. : 3403
Dato : 28/11-07
Ark nr. : 10 av 10

Denne kolonne er forbeholdt sensor.

I strukturerte metoden er man opptatt av å kartlegge hele omfanget først slik at man kan estimere tid og kostnad etterpå. Ofte er det jo nettopp slik at man bruker en strukturert metode fordi man har et stort sikkerhets/kontroll behov. Da må man vite så mye som mulig om omfanget først. Dette fører igjen til at slike prosjekter blir dyrere, fordi tiden strekkes ut. Her er det viktig at funksjonalitetskravene opprettholdes, det er derfor naturlig, ut fra trekanten, at kostnadene øker