

Forelesning IMT2243 17. Mars 2009

Dagens :

- Prosjektstatus – innlevering 20. mars kl 23:59
- Hvor er vi i emnet. Kort kommentar til viktige temaer og trender innen Systemutvikling som ikke er med i emnet
- To aktuelle temaer vi ser på idag
 - Open Source Software Development
 - Offshore Software Development

Pensum : Artsaml. 6, 7 og den "umerkede", mangler sider?

Kommentar til temaer innen systemutvikling som vi ikke ser på

- Koding / Construction (jfr. RUP) delen av SU-prosjekter
- Utviklingsplattformer (.NET, Eclipse ...)
- IT-sikkerhet i Systemutviklingsprosjekter
- Egenutvikling kontra Std.løsninger + Kunde/leverandør avtaler
- Mobile Løsninger og Grønn IT

Sentrale SU-temaer som dekkes i 3.kl emnet IMT3102 OOSU :

- Design Patterns
- SOA – Service Oriented Architecture
- Objektorientert Design (vi introduserer det bare i IMT2243)

Open Source Software Development

Open Source begrepet er i første rekke knyttet til et sett av kriterier for distribusjon av programvaren man har utviklet. Det er altså ikke bare snakk om at kildekoden skal være tilgjengelig. Aktører som legger ut sitt arbeid som Open Source legger dette tilgjengelig i form av at mottaker har frihet til å bruke og bearbeide videre.

Man kan etter å ha utviklet en programvare på tradisjonelt vis gjerne slippe det "ferdig" utviklede programmet ut som et Open Source prosjekt.

Noen Open Source-prosjekter tar steget videre og anvender en helt ny måte å kjøre selve utviklingen av programvaren på. Man velger her å involvere brukere av programvaren i selve utviklings- og testingsaktiviteten. Det er dette vi ser nærmere på.

Man "etablerer" en gruppe (ofte idealistiske) systemutviklere til å komme med sine bidrag i programvaren, samtidig med at man oppfordrer alle brukere der ute til å teste og komme med feilrapporter og forslag til utvidelser/ny funksjonalitet.

Litt historikk innen OSSD

- 1983 GNU-prosjektet startet av Richard Stallman. Mål om å lage et Unix-lignende operativsystem og distribuere det som fri programvare
- 1985 FSF Free Software Foundation stiftet av Richard Stallman
Støtte til universell frihet til å distribuere og endre programvare uten (copyleft). Fokuserer på fri som i frihet ikke som i gratis!
- 1991 Linus Torvalds lanserer sin ide om å utvikle et operativsystem og drar dermed i gang prosjektet som resulterer i Linux. I løpet av 2002 har anslagsvis 8000 årsverk av frivillig utviklingstid resultert i et operativsystem som benytter på 25 % av verdens servere.
- 1997 Eric Raymonds diskuterer i sitt manifest "The Cathedral and the Bazaar" det han mener er fundamentale forskjeller mellom utviklingsprosesser for henholdsvis tradisjonell kommersiell programvare (Cathedral model) og internett-basert samarbeid og fri informasjonsdeling i prosjekter som Linux og Fetchmail. Her diskuteres "The social context of Open Source Software".
- 1998 Open Source begrepet etableres (<http://www.opensource.org/docs/osd>)
- Hva finner vi om temaet på Wikipedia ?
http://en.wikipedia.org/wiki/Open_source_software_development

Felles karaktersitika ved sentrale Open Source prosjekter

Karakteristika hentet fra "Wide Open – Open source methods and their future potesial" av G.Mulgan m.fl
(<http://www.demos.co.uk/files/wideopen.pdf>)

- Transparency
- Vetting of participants only after they've got involved
- Low cost and ease of engagement
- A legal structure and enforcement mechanism
- Leadership
- Common standards
- Peer review and feedback loops
- A shared conception og goals
- Incrementalist – small players can still make useful contributions
- Powerful non-monetary incentives

Vi tar ikke for oss de ulike i distribusjonsformer og ulike lisensieringsavtalene som finnes innen fagfeltet. Se grunntakene som de fleste innen disse miljøene støtter på Open Source Initiative (link foran).

Karakteristika ved OSSD-prosesser

- Må ha et "startpunkt" i form av et eksisterende produkt eller en genial ide. Gjerne også et nettverk av kontakter innen Open Source miljøer.
- Hvor mange utviklere tror dere et gjennomsnittlig OSSD-prosjekt har ?
- Ulike utviklingsprosesser (A. Capiluppi m.fl):
 - Soloarbeid, en person eier og videreutvikler programvaren, men distribuerer den som Open Source
 - Soloarbeid med innslag av "eksterne patcher". Her er det en utvikler, men eksterne bidragsytere gir inn feedback, patcher og eventuelt forslag til utvidelser
 - Gruppearbeid med interne patcher. En lukket utviklergruppe som gjør all videreutvikling selv.
 - Gruppearbeid med reelle bidrag fra større utvikler- og brukermasser (Reelt OSSD).

Et "kroneksempel" på OSSD

Utviklingen av Linux ansees av mange som skoleeksemplet ikke bare på en velfungerende programvare distribuert som Open Source, men også som en reell OSSD utviklingsprosess der fordelene med å få engasjerte bruker- og utviklarmiljøer (User Community og Developer Community) er utnyttet til fulle.

Se : <http://no.wikipedia.org/wiki/Linux>

Kilder der dere finner mange eksempler på OSSD-prosjekt

Se <http://www.uversainc.com/download/top50.pdf> for eksempler på vellykkede og velfungerende Open Source Prosjekter.

Ellers er SourceForge og Freshmeat de viktigste kildene for å finne gode Open Source prosjekter. Merk at langt fra alle prosjektene er "aktive".

- <http://sourceforge.net/index.php>
- <http://freshmeat.net/>

Open Source – diskusjon :

Har dere noen eksempler på profilerte Open Source prosjekter ?

Sett opp mot de systemutviklingsmodellene vi har gjennomgått – hvilke karakteristika mener dere er sentrale innen OSSD ?

Hva finner dere igjen fra eXtreme Programming som kan være relevant i OSSD ?

Har dere noen betraktninger rundt viktigheten av responstid innen Open Source prosjekter ?

Hva er henholdsvis : User- og Developer Community og hvilken rolle spiller de i OSSD-prosjekter ?

Hvorfor tror dere folk bidrar som henholdsvis User og eller Developer ?

Hvorfor skulle du som HiG student vurdere å bidra ?

Open Source – kommentarer :

Kan dels med rette beskyldes for å være "feature-drevne" og uforutsigbare

Modularitet i produkt vesentlig. Design og arkitektur nøkkelområder

Kommunikasjon : epost, konfigurasjonsstyringsverktøy, wiki, web forum og prosjektplasser.

"Forking" – ganske vanlig at prosjekter forgreiner seg i løpet av livssyklus og eller skifter "eier". En god rettesnor å heller gå inn å prege videreutviklingen av eksisterende prosjekt fremfor å lansere "konkurrerende og overlappende" prosjekter.

Kommersielle aktører har for lengst entret arenaen
 - slipper eksisterende "lukket" programvare
 - kjører interne betalte utviklingsteam som forsøker å skape vinn-vinn situasjoner mellom selskap og brukermassen

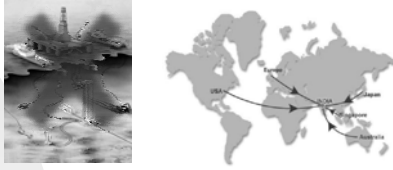
Prosessutfordringer innen OSSD

- "Startpunkt" i form av et eksisterende produkt eller en genial ide. Gjerne etablerte kontakter innen Open Source miljøer
- Basert på frivillighet. Må legge vekt på motivering av både utviklere og brukere
 - Beslutningshierarkier og åpenhet rundt dette viktig
 - Kort responstid
 - Kjerneutviklere med ekstremkompetanse på systemdesign
- Utviklerne er også brukere (gjør blant de avanserte brukerne), dermed blir fokus ofte snevert når det gjelder type programvare (jfr. kategorier av programvare)
- Klare skiller mellom stabile versjoner av programvaren og versjoner under utvikling

Nøkkelutfordringer hentet fra M.Levesques artikkel (artsaml. 7)

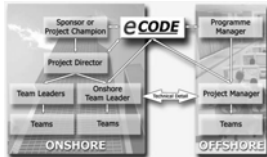
- Programvaren bærer ofte sterkt preg av å være utviklet av utviklerne for utviklere (en "lukket" verden)
- Har slitt med å få frem brukervennlige systemer særlig på Windows-plattformen. I de senere årene har utviklingen her vært meget positiv for eksempel med programvare som FireFox og OpenOffice.
- Dokumentasjonen på programvaren preges av å være laget av utviklere for utviklere
- "Feature-driven" utvikling
- Har vært klare tendenser til "Religions-krig" i forholdet mellom Open Source-miljøer og kommersielle aktører. Her er tendensen klart at man får oppmykning. Bl.a. har IBM distribuert utviklingsmiljøet Eclipse som Open Source.

Offshore Software Development



- "Offshore Software Development" er en trend i tiden innen måten man driver systemutvikling på. Dette er vår bransjes "svar" på globalisering.
- Store aktører og store utviklingsprosjekter kobler til seg kompetansen der man "får mest for pengene".
- Internasjonale firmaer etablerer miljøer med programmerings- og applikasjonsutviklings-kompetanse i lavkostland som India, Kina, Øst-Europa, men også i Irland, Israel og Canada

Offshore Software Development



- Utviklingsmodellen i slike prosjekter blir spesiell. Forholdet mellom kunde og leverandør blir mer kompleks. Leverandør har ofte organisasjonsenheter både Onsite og Offshore
- Internt hos leverandøren opplever man kulturskiller, tidssoneproblematikk og tildels språkproblematikk som kompliserende for utviklingen.
- Fordelen er at kan få tak i meget kompetente programmerere / testere / designere til en langt lavere pris

Offshore Software Development



- Nærheten mellom brukere og utviklere forsvinner.
- Ut fra detaljerte spesifikasjoner utvikler og tester Offshore-miljøene komponenter og delsystemer særlig innen større standardssystemer.
- Modellen kan benyttes til å utvikle alle typer systemer, men er kun hensiktsmessig i store prosjekter.
- Meget god skriftlig dokumentasjon og strenge kvalitetssikringsprosedyrer som etterleves er en forutsetning for å lykkes i slike prosjekter.
