KUNGL TEKNISKA HÖGSKOLAN
Royal Institute of Technology
Numerical Analysis and Computer Science

TRITA-NA-D0010 • CID-105, KTH, Stockholm, Sweden 2000

Temadag – iterativ utveckling och projektstyrning

Inger Boivie, Jan Gulliksen, Ann Lantz

CID
Centre for
User Oriented IT Design

Inger Boivie, Jan Gulliksen, Ann Lantz

Temadag – iterativ utveckling och projektstyrning

Report number: TRITA-NA-D0010, CID-105

ISSN number: ISSN 1403-0721 (print) 1403-073X (Web/PDF)

Publication date: November, 2000

E-mail of author: inger.boivie@hci.uu.se, jan.gulliksen@hci.uu.se, alz@nada.kth.se

URL of author: http://cid.nada.kth.se/

Reports can be ordered from:

CID, Centre for User Oriented IT Design
Nada, Dept. Computer Science
KTH, Royal Institute of Technology
S-100 44 Stockhom, Sweden
telephone: + 46 8 790 91 00
fax: + 46 8 790 90 99
e-mail: cid@nada.kth.se
URL: http://cid.nada.kth.se

Innehåll

Förord

Ett iterativt angreppssätt är en förutsättning för användarcentrerad utveckling. Men hur styr vi iterativa systemutvecklingsprojekt?

Systemutveckling, eller IT-utveckling, är i sig en mycket komplex verksamhet, svår att planera, styra och följa upp. CHAOS-rapporten[i] visar att endast vart sjätte systemutvecklingsprojekt lyckas i termer av leverans av ett användbart system inom de ursprungliga tids- och kostnadsramarna. Vattenfallsmodellen, grundad på tankar om planerbarhet och styrbarhet samt drömmen om den perfekta kravspecifikationen, döms ut av allt fler. Vi måste arbeta iterativt. Men om det är svårt att planera och styra "vattenfallsprojekt", hur ska vi då klara av att styra och planera en iterativ process, där osäkerheten är inbyggd redan från början? Vi ville bjuda in till en temadag på CID för att diskutera denna fråga.

Inger Boivie, TietoEnator AB

[i] www.standishgroup.com/visitor/chaos.htm

Upplägg

9.00-9.30 Ankomst, fika

9.30-10.30 Inledning och presentation av alla deltagare.
Nils-Erik Gustafsson, Ericsson Utvecklings AB
"Erfarenheter av utvecklingsarbete med och utan användarmedverkan - det var inte alltid bättre förr..."

10.30-12.00 Paneldebatt 1 -Vattenfall eller iterationer - sak samma?
Ordförande: Inger Dahlgren, TietoEnator
Deltagare:
Yvonne Dittrich, Karlskrona/Ronneby Universitet
Henrik Öbrink, Init
Frågor att diskutera:

- Vad innebär de olika begreppen vi använder? Vad är iterativ utveckling, inkrementell? Finns det någon verklig skillnad?
- Är det så att vi alltid utvecklar iterativt - men mer eller mindre kontrollerat?
- Är det så att vi alltid utvecklar i vattenfall - men stegen har blivit kortare och fler?

12.00-13.00 Lunch

13.00-14.30 Hur styr vi ett iterativt projekt - ett fiktivt praktikfall- föredragare: Inger Boivie, TietoEnator
Gruppdiskussioner - fika under tiden

14.30-16.00 Paneldebatt 2 - Iterativ utveckling - på andra sidan leveransdatum
Ordförande: Sara Eriksén, Karlskrona/Ronneby Universitet
Deltagare:
Ted Challis, Botkyrka kommun
Per-Magnus Skoogh, DSDM-konsortiet/Open World Management
Nils-Erik Gustafsson, Ericsson Utveckling AB
Frågor att diskutera:

- Hur förhåller vi oss till iterativ utveckling sedd i ett längre perspektiv - över systemets/produktens hela livslängd?
- Hur styr vi den iterativa utvecklingen av systemet/produkten över hela dess livslängd?

Sammanfattning

Dagen inleddes med ett föredrag av Nils-Erik Gustafsson. Han diskuterade användarmedverkan i olika former och delade med sig av sin mångåriga erfarenhet av systemutveckling och användbarhetsarbete.

Debatterna rörde framförallt iterativ utveckling kontra vattenfallsmodellen, samt huruvida färdiga modeller, t ex RUP och DSDM, erbjuder lösningen på våra problem.

Den första debatten inleddes med ett försök att definiera de olika begrepp vi rör oss med, vattenfall, iterativ utveckling samt evolutionär och inkrementell utveckling. Den stora skillnaden i de olika begreppen ansågs vara de synsätt som styr dem, och de antaganden som ligger till grund för dem. Detta medför skillnader i hur vi kommunicerar och interagerar inom ett projekt snarare än i de aktiviteter vi genomför.

Debatten fortsatte med en diskussion om nackdelar och fördelar med vattenfallsbaserad utveckling jämfört med iterativ utveckling. Vissa menade att den förra modellen är att föredra emedan den är lättare att planera och styra. Andra menade att den baseras på ouppnåeliga drömmar om den perfekta kravspecifikationen, och att vi alltid hamnar i lägen där vi måste iterera. Det är därför bättre att medvetet välja en iterativ modell.

Dock kompliceras situationen av att affärsmodellen sällan tillåter en iterativ utvecklingsmodell. Det finns en tvekan hos beställaren att fatta beslut om utveckling med mindre än att han/hon får en garanti om vad resultatet blir. Vattenfallsmodellen erbjuder skenbart denna garanti.

Som avslutning på den första debatten diskuterade vi huruvida vi verkligen behöver modeller – dock var alla överens om att de behövs. Man behöver gemensam plattform, gemensamma värderingar, språk och processer i all utveckling. Vad modellen heter spelar dock mindre roll.

Debatt två fokuserade på vad som händer efter leveransdatum. Många menade att det är först då vi får veta om vi lyckats uppfylla alla krav, både uttalade och outtalade. Oundvikligt är också att verksamheten förändras av det nya systemet – vilket i sin tur förändrar kraven på det. Detta medför att systemet inte är klart efter leverans. Ett problem i denna process är att utvecklingsorganisationen oftast avvecklas och nya personer tar över och vidareutvecklar systemet. För att inte förlora kontinuitet krävs användarmedverkan och bra dokumentation av systemarkitekturen.

Vissa menade att man bör se framförallt webbutveckling som en process, bestående av ett antal små projekt. Det ena tar vid där det andra slutar och webbplatsen/systemet blir aldrig klart. Ett problem i sammanhanget är då att mottagarorganisationen inte vill hamna i beroendeställning till en viss leverantör.

Avslutningsvis kom vi tillbaks till diskussionen om de färdiga modellerna – kan de rädda oss från alla problem? Kontentan av den diskussionen var att, som nämnts ovan, namnet och innehållet i modellen inte är så viktigt, som det faktum att man har en gemensam modell i projektet. Dessutom tillför de färdiga modellerna ett värde i det att de faktiskt försöker lösa vissa problem

– ett antal personer har faktiskt satt sig ner och tänkt igenom problem som är gemensamma i många projekt. De ger oss definitioner på begrepp som vi tidigare inte varit överens om. De är ett uttryck för en växande sjukdomsinsikt i branschen. De är dock aldrig “the silver bullet”.

Dagen innehöll också en gruppdiskussion där uppgiften var att diskutera ett antal frågor angående iterativ utveckling, givet ett fiktivt praktikfall. De flesta grupperna skissade på modeller för utvecklingen.

Erfarenheter av utvecklingsarbete med och utan användare – det var inte alltid bättre förr

Inledande föredrag

Dagen inleddes med ett föredrag av Nils-Erik Gustafsson, Human Factors Expert, Ericsson Utveckling AB. Nils-Erik berättade om sina erfarenheter av att arbeta med och utan användare i system-utvecklingsprojekt. Han beskrev hur inställningen till användar-medverkan och attityder har förändrats genom åren och illustrerade med exempel från några av de många olika projekt han deltagit i.

Hur var det förr

Nils-Erik beskrev några av de tidiga projekten där han deltagit. Dessa följde en traditionell vattenfallsmodell – med specifikationer för krav, funktioner, implementation, test, etc. Utrymmet för användar-inflytande i processen var begränsat.

Användar-medverkan – när, var och hur

Nils-Erik framhåller vikten av att involvera användare, riktiga användare, i processen. Dock bör man inte ta in användare i systemutvecklingsprojektet för att "tömma dem" på information om hur de arbetar. Det är fel att låta en användare lämna sin arbetsplats under en lång period för att delta i utvecklingsprojektet. Han/hon tappar snabbt förmågan att beskriva hur man verkligen arbetar på arbetsplatsen. Det bästa är att själv gå ut till användarna och deras arbetsplats – sitta med och iaktta. Genom att vara kvar i sin arbetsmiljö får användaren hjälp med att beskriva och komma ihåg, från ledtrådar i själva omgivningen.

Biljettbokningen

Som exempel på ovanstående arbetssätt beskrev Nils-Erik ett projekt med syftet att utveckla ett nytt biljettbokningssystem. Utvecklingsprojektet hade fokuserat på biljettexpeditören och inte på den resande. En biljettexpeditör hade deltagit i utvecklingsarbetet, och efter 1,5-2 år som expert visste han inte längre hur man egentligen arbetade med biljettförsäljning. Han hamnade i en omöjlig situation där han varken var biljettexpeditör eller systemutvecklare. Han blev också gisslan i projektet som "användarrepresentanten".

Nils-Eriks uppdrag var att göra systemet mer objektorienterat. Han ignorerade i princip all dokumentation och gick ut till biljett-försäljningskontoren. Med papper och penna i hand observerade han hur man arbetade och "tittade över axeln" på resenärer. Under pauser i arbetet passade han på att fråga saker. Han skissade också på nya lösningar på plats som han direkt kunde få återkoppling på.
Att filma gick inte, eftersom det kräver deltagarnas medgivande och dessa ofta var så stressade att det inte fanns plats för någon sådan diskussion.
Resultatet blev en helt ny lösning. Nils-Erik föreslog förändringar i grundförutsättningarna, t ex att slå ihop färd- och platsbiljett. Han strukturerade också om informationen på skärmen efter den dialog biljettexpeditören för med resenären. "Vart ska ni åka, när, hur?"

Prototyper är bra

Prototyper är ett ypperligt verktyg för att generera krav. Man kan inte skriva ett dokument som beskriver alla krav. Det är först när man ser en prototyp som man kan börja diskutera runt behov och krav. Med prototyper kan man också börja testa informellt och tidigt. I brist på annat kan man testa inom organisationen med surrogatanvändare, dvs personer som inte ska använda systemet, men som ändå "liknar" de riktiga användarna vad gäller t ex domänkunskap och bakgrund.

För att signalera att något är under utveckling och inte alls färdigt använder Nils-Erik vad han kallar "pilsnerlinjer", dvs linjer dragna för hand. Man kan också bygga in återkopplingsfunktioner i systemet så att användarna lätt kan skicka in kommentarer.

Nils-Erik beskrev ett projekt där man skulle utveckla en inomhusbankomat. Han fick till uppgift att bestämma hur långt det skulle vara mellan knapparna och kunde, med hjälp av prototyper, snabbt komma tillbaka med ett resultat. Resultatet i sig var inte det viktiga utan det faktum att han lämnade ett snabbt svar. På så vis vann han organisationens förtroende och kunde ta upp viktiga användbarhetsfrågor.

Labb eller inte labb

Nils-Erik berättade om sina erfarenheter från IBMs användbarhetslabb. Det första i Sverige. Man hade falska speglar och utrustning för att spela in användarens interaktion och reaktioner. Dock kunde man konstatera att labbtest är tunga och tidskrävande, det tar t ex mycket lång tid att analysera det inspelade materialet (ca 3-5 ggr inspelningstiden). Tendensen blev därför att man sköt på testerna, och användarmedverkan, till sent i utvecklingsprocessen.

Nils-Erik menar att det går snabbare med expertgranskningar och informella användartester. Ett välvalt citat från en användare säger ofta mer än en tio- eller tjugusidig rapport.

Labb kan vara bra för säkerhetskritiska system, men för andra system är det inte nödvändigt med den formen av tester, och därmed svårt att rättfärdiga kostnaden för ett labb. Produkterna blir orimligt dyra om de ska vara perfekta. Vi måste lägga oss på en nivå som är tillräckligt bra för att hamna på en rimlig prisnivå.

Vad är ett labb

En diskussion uppstod om hur vi definierar användbarhetslabb. Idag kan man ta med en videokamera och filma ute på fältet i den naturliga användningsmiljön, sk mobila labb.

Någon föreslog att användbarhetslabb skulle definieras som delat arbetsutrymme (shared workspace) – dvs ett utrymme med utrustning för videoanalys och prototyping och för kreativa design-workshops.

Nils-Erik menade att man ofta behöver vare sig labb eller videokamera för att komma ihåg de värsta användbarhetsbristerna. De är så iögonfallande att man inte glömmer dem i första taget. Å andra sidan ger en videofilm möjligheten för flera att ta del av resultatet och bedöma det. Detta öker graden av objektivitet.

Användbarhetsmål

Det krävs inte heller labb för att arbeta med användbarhetsmål. Man kan t ex mäta tiden det tar att genomföra en uppgift med ett vanligt stoppur. Man kan också konstatera huruvida användaren klarade av en uppgift eller inte utan loggningsutrustning. Ofta krävs inte större precision än så.

Låt användaren bestämma

Nils-Erik menade att vi inte ska göra systemen för smarta, dvs att de löser uppgifter eller tillhandahåller information utan att användaren ber om det. Vid tveksamma fall är det alltid bättre att låta användaren själv bestämma. Som exempel tog han bankomaten – där hade man i en tidig lösning valt att presentera saldot automatiskt. Detta är dock en högst privat uppgift som användaren kanske inte vill att andra ska se. Det är därför bättre att låta användaren själv bestämma om han ska titta på saldo eller inte.

Paneldebatt 1 – Iterativ utveckling – vad är det?

Inledning

Deltagare

- Panelordförande: Inger Dahlgren, TietoEnator
- Paneldeltagare:
  - Henrik Öbrink, Init
  - Yvonne Dittrich, Karlskrona/Ronneby Universitet

Vad pratar vi om

Inger Dahlgren inledde med att diskutera vad alla de begrepp vi rör oss med betyder. Är de egentligen olika etiketter på samma sak? Är t ex iterativ utveckling samma sak som vattenfallsmodellen fast kortare och snabbare? Eller är vattenfallsmodellen egentligen iterativ utveckling, där första iterationen utgörs av utvecklingsprojektet som sedan följs av ett antal iterationer under underhållsfasen?

Sammanfattning

Debatten rörde framförallt följande frågor

- Definitioner – finns det någon reell skillnad mellan de olika begreppen. Allmänt ansågs att den stora skillnaden finns i förhållningssätt och vår syn på utvecklingsprocessen.
- Vattenfall kontra iterativ utveckling. Vissa menade att vattenfall är lättare att planera och styra, samt billigare. Men den kan enbart användas vid stabila förutsättningar, både vad gäller krav och teknik. Andra menade att vi alltid hamnar i situationen där vi måste iterera, oavsett om vi planerat det eller inte. Det är därför bättre att medvetet välja iterativ utveckling.
- Hur använder vi modellerna? Använder vi dem överhuvudtaget? Behöver vi dem? Det senare kunde vi vara överens om - vi behöver dem, men vi måste situationsanpassa val och utveckling av modell. Problemet är dock att väldigt få kan använda modellerna, därmed försvinner också steget där man anpassar dem.
- Kan byggindustrin bättre? Kanske. Men våra förutsättningar skiljer sig i komplexitet, material, komponenter och synliga resultat av processen.

Avslutning

Avslutningsvis jämförde Inger Dahlgren förändringarna i system-utvecklingsprocessen med de förändringar som sker runt om oss i samhället. Vi har ett behov av modeller, det okända och osäkra gör oss frustrerade och oroliga. Vi vill kunna kontrollera processen både före, under och efter men saknar redskap för att styra och följa upp. Vi vill ha strukturer – men systemutvecklingen kanske bara följer samhällsutvecklingen i övrigt. Det kanske är så att vi inte kan leva och arbeta i så fasta strukturer som vi hittills gjort. Vi måste anpassa oss i mycket högre grad efter situationen – system, användare, organisation, etc.

Definitioner

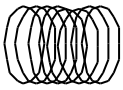
Definitioner nödvändiga

Vi inledde debatten med ett försök att definiera vissa begrepp vad gäller utvecklingsprocessen, i syfte att skapa en gemensam plattform för diskussionen. Vi nådde inte ända fram, men nedanstående definitioner diskuterades.

Vattenfall

Vattenfallsmodellen fokuserar på funktioner. Varje delsteg i modellen är väldefinierat och genererar ett dokument eller artefakt som förs vidare till nästa steg. Vattenfall bygger på att ett problem kan analyseras "klart" för att sedan lösas.

Iterativ utveckling



Att iterera innebär att gå tillbaka och göra om samma sak gång på gång. Man skulle kunna säga att stegen i vattenfallsmodellen blir kortare. Men till skillnad från vattenfallsmodellen ställs inga krav på att analys, krav, system, etc ska vara kompletta i de första iterationerna. Tester och utvärderingar utförs på prototyper och erfarenheterna tas till vara och förs tillbaka till utvecklingsprocessen. Iterativ utveckling är ingen garanti för att man faktiskt tagit till vara användarnas reaktioner vid verklig användning.

Iterativ utveckling kan illustreras med en spiral där man upprepar processen att identifiera, planera, genomföra och granska/utvärdera.

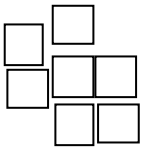
Iterationer finns på olika nivåer i projektet – dels itererar vi vår lösning inom projektet så att den stämmer med målet. Dels itererar vi tillsammans med användare. I det senare fallet vill vi inte ha ett fast mål – iterationerna syftar till att förändra målet. Det är denna delen av processen som är den svåra och utmanande att arbeta med. Hur utvecklar vi systemet/produkten samtidigt som vi utvecklar målet?

Evolutionär utveckling

I evolutionär utveckling fokuserar man i högre grad på affärsprocessen. Vilken nytta medför IT-stödet, vad vill man uppnå? Nyttoaspekten ska vara styrande. Även här genomförs utvecklingen i små steg och man bestämmer redan från början vilka steg som ska genomföras. Varje steg ska resultera i en levererbar, fungerande del av systemet som kan sättas i produktion ute hos användarna. Deras reaktioner kan sedan återföras till utvecklingsprojektet. I evolutionär utveckling återförs alltså den riktiga användningssituationen till utvecklingsprocessen – och man utnyttjar den inlärningsprocess som alltid blir resultatet av ny teknik.

Evolutionär utveckling kan illustreras med en symbol som ändrar form över tiden, dvs man närmar sig något steg för steg. Den slutliga formen är inte känd från början.

Inkrementell utveckling



I inkrementell utveckling arbetar man med övergripande krav och utgåveplanering. Man tar fram något som kan sättas i produktion och användas för att samla in reaktioner. Den första utgåvan är inte perfekt, men går att använda. Inkrementell utveckling kan illustreras med en samling smårutor, där man beslutar om vilken helhet man vill skapa och sedan släpper bit för bit.

Vissa menade att produkten är inkrementell, medan processen är iterativ/evolutionär.

Användar - medverkan

Iterativ utveckling förutsätter att användarna finns med i processen. Det är meningslöst att iterera om man inte inhämtar användarnas reaktioner på systemet under gång. Detta ställer i sin tur höga krav på organisationen – den måste tillåta och uppmuntra “snurrorna”.

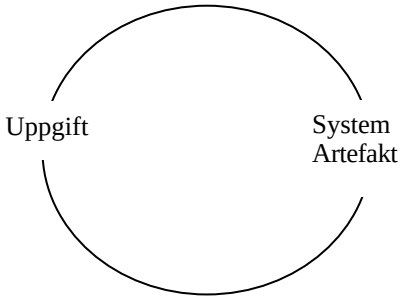
Skillnader

Skillnaderna ligger framförallt i synsätt, attityder och förhållningssätt till systemutveckling. Den största skillnaden kanske ligger i hur man förhåller sig till förändringar.

I vattenfallsmodellen finns en dröm om och en strävan efter planerbarhet och den perfekta kravspecifikationen – i iterativ utveckling finns inte detta. Det medför helt andra sätt att kommunicera och interagera med varann i processen.

Illustration av skillnader

Bilden nedan illustrerar skillnaden mellan iterativ utveckling och vattenfallsmodellen



Om utveckling av både uppgift och system ingår i utvecklings-processen är den troligen iterativ. Om bara systemutveckling ingår medan utveckling av arbetsuppgifterna ingår i underhållsfasen är processen troligen vattenfallsbaserad.

Att iterera är att bygga kunskap

Någon menade att iterativ utveckling inte handlar om att upprepa samma saker gång på gång, eller att köra vattenfallsstegen gång på gång. Det handlar om att bygga kunskap inom en viss domän. Iterativ utveckling innebär att en grupp människor ägnar sig åt att bygga upp kunskap som de ger ett synligt uttryck för, t ex i form av en prototyp. Sedan går de in i nästa iteration, som kan omfatta helt andra aktiviteter, men som alltid har syftet att bygga vidare på den kunskap den första iterationen skapade. Den verkliga uppgiften är inte att upprepa, utan att bygga upp kunskap.

Evolution och revolution

Ytterligare en dimension togs upp, nämligen frågan om evolutionär kontra revolutionär prototyp-utveckling.

- Evolutionär utveckling innebär att man aktivt återanvänder så mycket som möjligt från prototyp till färdigt system.
- Revolutionär utveckling innebär att man kastar bort prototypen och börjar om när man börjar utveckla det slutliga systemet. En annan syn på revolutionär utveckling är att lösningen blir en helt annan en den man hade tänkt från början. En uppgiftsanalys kanske visar att det som behövs är inte ett hjul utan en skida.

Vattenfall kontra iterativ utveckling

Vattenfall eller iterationer

En stor del av debatten handlade om vattenfallsmodellen kontra iterativ utveckling. Synen på vattenfallsmodellens användbarhet skiljde sig mycket inom gruppen. Är den bra, lätthanterlig, lättstyrd eller helt omöjlig att använda? Är det så att vi alltid itererar, fast vi kallar det förändringshantering och formaliserar det så att det blir ohanterligt och tungt?

Det omöjliga vattenfallet

Vissa menade att vattenfallsmodellen egentligen är att föredra eftersom den är lättare att planera och styra än iterativ utveckling. Dock är den endast användbar i "stabila" projekt. Om teknik, kravbild, etc är instabila bör man absolut inte använda vattenfallsmodellen.

Andra invände att även om man väljer vattenfall tvingas man iterera under processen, eftersom verkligheten *alltid* förändras under tiden. Det är bättre att medvetet bestämma sig för att arbeta iterativt – då kastar man bort två ouppnåeliga drömmar:

- drömmen om den frysta kravspecifikationen
- drömmen om att man kan sätta ihop en grupp med människor som kan komma fram till det enda sanna svaret

Det är alltid så att man i början av ett projekt inte vet någonting , man måste ta vara på det faktum att man utvecklar sin kunskap under resans gång.

Vattenfall används ändå

Trots alla invändningar mot vattenfallsmodellen så finns och används den förhållandevis flitigt inom industrin. I många projekt strävar man efter frysta kravspecifikationer och uttalade mål. En orsak till detta är att beslutsfattare – de som betalar för utvecklingsprojekten – vill veta vad de får för pengarna innan de fattar beslut om att spendera dem. Med vattenfallsmodellen kan man i varje fall ge sken av att kunna specificera detta.

Billigt eller dyrt

Är vattenfallsmodellen billigare än de andra, eftersom den är lättare att planera och styra? Eller är det så att vattenfallsmodellen kan leda till att totalkostnaden sett över systemets/produktens hela livslängd blir större eftersom användningskostnaden riskerar bli högre.

Ett problem är att affärsprocessen ofta inte tillåter iterativ utveckling. Utvecklingskostnaden får en etikett och betalas ur en påse, medan användningen får en annan etikett och betalas ur en annan påse. Dock borde det gå att sälja in en annorlunda utvecklingsprocess om man kan visa att slutresultatet blir bättre.

Börja om

Man skulle kunna se vattenfallsmodellen som en inkrementell eller iterativ process, där den första delen omfattar själva utvecklingsprojektet fram till och med införande. Resterande inkrement eller iterationer utgörs av vidareutveckling och felrättning under underhållsfasen.

Den stora skillnaden och nackdelen gentemot verkligt iterativ utveckling är dock att det är olika organisationer som ansvarar för utveckling och underhåll. De personer som deltar i utvecklingen går ofta vidare till andra utvecklingsprojekt. Personalen byts ut mellan de olika iterationerna – detta gör att man tappar bort all kunskap mellan dem. Om man istället itererade skulle man kunna behålla personal och kunskap mellan omgångarna.

Hur använder vi modellerna

När, var, hur?	Det fanns inga enkla svar på när, var och hur vi ska använda modellerna. En viss modell kan inte sägas vara bättre eller sämre än en annan. De måste sättas in i sitt sammanhang, och anpassas efter situationen. Använd dem med sunt förnuft, och se dem som resurser/förhållningssätt snarare än kokböcker.
Ursprung	Det hävdades att det inte går att bedöma modellerna som "bra" eller "dåliga". Man måste ta hänsyn till det sammanhang där en viss modell har utvecklats. Vattenfallsmodellen, har t ex sitt ursprung i anbudsförfarandet, där alla aktörer ska "tävla" på samma villkor. Det innebär att alla måste veta vad man förväntas bygga. Idag används system/produkter till andra saker än på den tiden vattenfallsmodellen utvecklades, då krävs också andra modeller för framtagningen av dem.
Situations-anpassning	Val av modell och anpassning av den måste ske med hänsyn tagen till situationen där resulterande system/produkt ska användas. Vi rekommenderades att läsa "The Computer Reaches Out: The Historical Continuity of Interface Design"[ii] av Jonathan Grudin som diskuterar hur forskningen och fokus kring användargränssnitt har förändrats över tiden, från 1950-talet till nutid. En jämförelse gjordes med ledarskap inom försvarsmakten – från början användes auktoritärt ledarskap och det visade sig inte fungera. Då gick man över till demokratiskt ledarskap, som inte heller fungerade i alla situationer. Lösningen blev situationsanpassat ledarskap. På samma sätt måste modellen situationsanpassas. Definitioner och modeller är fint och bra, men när man "sitter i skiten" då väljer man den lösning man vet fungerar och som man har resurser och tid till.
Förstudie	Förstudien är mycket viktig. Om man väljer t ex vattenfallsmodellen för sin utveckling och sedan misslyckas, beror det på att förstudien fallerat. En väl genomförd förstudie och ett medvetet val av modell är en förutsättning för att lyckas. I många fall kan kunden/beställaren ha bestämt sig för en viss modell därför att den är "på modet" – då krävs det en väl genomförd förstudie samt ett visst mått av civilkurage för att upplysa om att den modellen inte passar i den givna situationen.

[ii] The Computer Reaches Out: The Historical Continuity of Interface Design, Evolution and Practice in User Interface Engineering / Jonathan Grudin, Proceedings of ACM CHI'90 Conference on Human Factors in Computing Systems 1990 p.261-268

Ingen följer modellerna

Ett problem med färdiga systemutvecklingsmodeller är att ingen följer dem – i praktiken. Man använder dem för att kommunicera med andra, ställa krav, handla upp konsulter, etc. I många organisationer gör man saker man egentligen inte kan, då är det lätt att bli förblindad av nya fina modeller med tjusiga namn och förpackningar. De som är ansvariga för upphandlingen ser ofta inte mer än så. Det har helt enkelt gått mode i modeller.

I praktiken, då modellen verkligen ska användas, krävs det alltid en anpassning av den. Många modeller löser detta genom att ha ett första steg som innebär anpassning av modellen. Men eftersom ingen följer modellen genomförs aldrig denna anpassning.

Vi behöver modeller

Trots detta ansågs modeller nödvändiga. Man måste följa vissa regler när man utvecklar. Man kan t ex inte införa förändringar hur som helst. Modellen ger de involverade ett visst mått av trygghet. Det är bättre med en dum, stel modell, om alla vet hur den fungerar, än ingen alls.

Dock måste alla modeller användas med sunt förnuft och måtta. De ska förse oss med en ledstång när vi springer nerför trappan och riskerar att snubbla. Ett problem är dock att alla tror att man kan släppa ledstången väldigt tidigt. Det slutar i värsta fall med att “man springer i olika trappor eller rent av åker hiss istället”. Man slutar helt enkelt kommunicera.

Modellen är en resurs

Vissa menade att vårt synsätt på modellen som en ledstång, eller som en “guide” för personerna i projektet begränsar oss. Vi borde se dem som en resurs - ett ramverk – som styr vårt förhållningssätt och vår syn på utvecklingsmodellen snarare än kokböcker för vilka aktiviteter vi ska genomföra. Börja med en idé om hur ni vill se på er modell och utveckla den sedan allt eftersom arbetet framskrider. Viktigt är dock att alla är involverade i utvecklingen av modellen.

Kan byggindustrin det vi inte kan?

Byggindustrin

Den oundvikliga jämförelsen med byggindustrin togs upp. Likheterna med mjukvaruutveckling är många. Generellt anses byggindustrin klara av att leverera till utsatt tid och kostnad med utlovad kvalitet. Invändningar fanns dock mot detta:

- Den positiva bilden av byggindustrin är överdriven. Förseningar, otydligheter, felaktigheter, fasta priser som inte alls är fasta, iterationer och undermålig kvalitet förekommer även här. I stor utsträckning, trots att det borde finnas rutin.
- Förutsättningarna är annorlunda. Vid ett bygge är processen och resultatet av den synliga under resans gång – om en snickare sätter upp en vägg, ser elektrikern den. I mjukvaruvärlden är resultatet av många aktiviteter osynliga för andra ända till slutet av processen. De blir synliga först i integrationstesterna.
- "Materialets" egenskaper är annorlunda. Kod är betydligt mer komplext än gips och betong. Istället bör vi jämföra oss med t ex flygindustrin, där materialets komplexitet är på samma nivå som många utvecklingsprojekt inom mjukvaruvärlden. Många IT-projekt är otroligt komplexa.
- I byggindustrin använder man färdiga komponenter – i mjukvaruvärlden bygger vi nytt hela tiden.

Paneldebatt 2 – Iterativ utveckling på andra sidan leveransdatum

Inledning

Deltagare

- Panelordförande: Sara Eriksén, Universitetet Ronneby/Karlskrona
- Paneldeltagare:
  - Ted Challis, Botkyrka kommun
  - Per-Magnus Skoogh, DSDM-konsortiet/Open World Management
  - Nils-Erik Gustafsson, Ericsson Utveckling AB

Sammanfattning

Debatten berörde främst följande frågor

- Vad händer efter leveransen – det är när systemet satts i drift som den egentliga testen av det börjar. Vad händer då? Hur tar vi hand om vidareutveckling, frågor som kommer via "helpdesk"? Hur skapar vi kontinuitet i processen? Ska vi iterera före eller efter leverans?

- Kan de färdiga modellerna frälsa världen? Vi diskuterade för och nackdelar med färdiga modeller, framförallt RUP och DSDM. Dock konstaterade vi att mottagarsidan är tämligen ointresserad av vilken modell som används. De har helt enkelt ett problem som de vill ha löst!

Avslutning

Sara avslutade med en kort diskussion runt standardisering och generalisering. Det finns en svårighet att balansera mellan dessa två. Hur kan man rationalisera processen att specialsy applikationer för specifika organisationer men ändå ha en standardlösning? Många av de här problemen är högst relevanta för diskussionen om vad som händer efter leveransdatum. Man kan inte driva det som ett systemutvecklingsprojekt in absurdum. Hur ska man t ex arbeta med "helpdesk"?

Vad händer efter leveransen?

Systemet i användning

Det är när systemet sätts i drift och användningen börjar på allvar som det intressanta inträffar. Det är först då man vet om man uppfyllt alla krav, både uttalade och outtalade sådana. Dessutom kommer införandet av det nya systemet att medföra förändringar i verksamheten – vilket i sin tur förändrar kraven på systemet. Användarna må vara nybörjare i början, men de lär sig snabbt, därför måste systemet och tilläggsinformationen vara utformat så att det gynnar vana användare. “Jag vill jobba, inte köra mus”.

Användare är ett kreativt släkte – i många situationer vill de ha redskap till att göra saker som varken systemutvecklarna eller de själva har en aning om när systemet byggs. Vi bör därför lämna vår strävan att bygga det perfekta systemet och istället bygga verktygslådor som användarna själva anpassar.

Projekt i process

Vissa saker kan inte upptäckas förrän systemet satts i drift. Vid t ex webbutveckling släpper man sin webbplats och låter användare testa den under en period, sedan vidareutvecklar man den, modifierar och lägger till nya tjänster. Den blir aldrig klar – därför är det mer lämpligt att se webbutveckling som en process, där små projekt avlöser varann.

Om man, som webbutvecklare, ska göra en bra leverans – kan man inte släppa den efter lanseringen. Å andra sidan vill inte kunden vara uppbunden till en enskild webbkonsult. Kunden kanske har köpt ett antal underkonsulter som arbetar tillsammans som en grupp fram till lanseringen. Problemet uppstår då efter lanseringen – hur ska man kunna säkerställa fortsatt kvalitet efter leverans? Liknande problem finns vid traditionell systemutveckling – systemet ska levereras till en miljö – och ska ses som en del i denna miljö. Hur kan utvecklingsorganisationen säkerställa kvaliteten i denna miljö? Man kan inte släppa systemet efter leverans.

Användarna skapar kontinuitet

Det vanliga vid leverans är dock att projektet tar slut. I de fall man fortsätter med vidareutveckling är det ofta andra personer som tar över och startar från ruta noll.

Ett sätt att skapa kontinuitet är att involvera användare. Om användarna är med och utvecklar systemet i en iterativ process, ökar chanserna att projektet lyckas leverera något användbart. Användarna blir då kunskapsbärare i den fortsatta utvecklingen av systemet. Dessutom krävs dokumentation av systemarkitekturen, på hög nivå.

Användarna en resurs

I den ideala processen ses användarna som en resurs för systemutvecklarna – istället för tvärtom. Frågor och önskningar som kommer in via “helpdesk” blir underlag för vidareutveckling. Användarmedverkan och iterativa processer kräver dock en mogen mottagarorganisation – och utbildade, erfarna användare.

Iterationer före eller efter leveransdatum

Vi bör ändå sträva efter att få ett så bra system som möjligt redan innan systemet satts i drift, och inte låta vidareutvecklingen ta hand om all anpassning till den verkliga användningen – att iterera det färdiga systemet istället för att iterera lösningsförslagen på ett tidigt stadium.

Om man itererar under hela processen, t ex släpper en ny prototyp i veckan, styr man projektet hela tiden. Detta ökar avsevärt chanserna att lyckas vid leverans. Man får också bieffekten att man säljer in systemet – man skapar en känsla av delaktighet hos användarna. De får vara med och påverka. I värsta fall kan det dock vara en falsk känsla.

Konsulter kontra dataavdelning

Är iterativ utveckling efter leveransdatum möjlig endast om man har en egen dataavdelning som vidareutvecklar och drifthåller systemet? Hamnar man i beroendesituation om man låter konsulter ta hand om den delen?

Felet med detta resonemang, menade någon, är att man ser mjukvara som en produkt, en sak som man köper. Men det är egentligen en uppsättning tjänster, som systemet levererar till mottagaren. Därför är det naturligt med en fortsättning av relationen mellan utvecklingsorganisationen och mottagarorganisationen, oavsett om det är externa konsulter eller dataavdelningen som stått för utvecklingen. Man bör sträva efter en långsiktighet.

Kan färdiga modeller frälsa oss?

Moderna systemutvecklings-modeller

Debatten fokuserade i mångt och mycket runt moderna system-utvecklingsmodeller, framförallt Dynamic Systems Development Method (DSDM) och Rational Unified Process (RUP).

DSDM är ett exempel på en iterativ process, som man, enligt DSDM-representanten, kan "hålla i handen". Den är framförallt anpassad för projekt med stor osäkerhet, t ex vad gäller kravbilden. Processen baseras bl a på iterationer av prototyper.

The silver bullet

Om det nu finns färdiga modeller som, enligt förespråkarna, löser alla problem – varför diskuterar vi då problem? Vi vet att endast var sjätte IT-projekt lyckas, i termer av att leverera något användbart på tid och pengar (CHAOS-rapporten[iii]). Det finns inga bevis på att de nya modellerna har förbättrat denna dystra statistik, annat än den information som leverantörerna av modellerna själva presenterar.

Många menade dock att de färdiga modellerna har ett värde i att de erbjuder något att "hålla sig i". Om man t ex har 50-100 personer i ett projekt måste man ha gemensamma processer, värderingar och termer. Definierade modeller är bra, men måste ses som ramverk. De måste anpassas efter situationen – detaljerade checklistor fungerar inte. *Vad* modellen heter spelar dock mindre roll.

Någon påpekade att färdiga modellers popularitet kanske kan vara symptom på att alla är frustrerade över att de som finns inte fungerar. Snarare än att den färdiga modellen skulle erbjuda en överlägsen lösning.

Sjukdomsinsikt

Någon menade att man inte ska stirra sig blind på om en modell är bra eller inte. Det intressanta med t ex RUP och DSDM är att de tar vissa problem på allvar. Det finns personer som lägger tid på att försöka lösa dessa problem – detta i sig är ett skäl till att hålla ett öga på modellerna. De borde innehålla åtminstone vissa delar som kan vara till nytta i verkliga projekt.

Modeller skapar också ett gemensamt språk, de ger definitioner av begrepp som vi inte varit överens om tidigare. De är helt enkelt uttryck för en växande sjukdomsinsikt i branschen.

Ledstång eller halleluja

Ett problem med färdiga modeller, som tidigare diskuterats, är att de ska fungera som ledstänger, men de används ofta till att åka hiss till rent halleluja-mässiga höjder.

Det finns bra saker och visdom i modellerna, men de är aldrig "the silver bullet". Det går ibland lite fel, ibland lite rätt, men det går alltid mycket bättre än om man skapar en egen modell på sin kammare.

[iii] www.standishgroup.com/visitor/chaos.htm

Yrkesmässighet	Det är personerna i projektet som skapar förutsättningar för att lyckas, inte modellerna. Komplex verksamhet låter sig näppeligen uttryckas i regelverk. Avancerad systemutveckling kräver yrkesskicklighet som man lär sig genom erfarenhet. Om man behärskar ett ramverk kan man göra avsteg på ett intelligent sätt.
Öppna standarder	Vad är en öppen standard. Hur öppet är något som är öppet? Om det verkligen är öppet ska det vara tillgängligt och förståeligt för alla – vilket den här typen av modeller knappast är.

Gruppdiskussioner

Upplägg

Form

Åhörarna delades in i olika grupper, var och en med fem till sex deltagare. Diskussionen varade i ca 1 timme och resultatet presenterades sedan kort inför alla. Som underlag för diskussionen fanns ett “fiktivt praktikfall”.

Det fiktiva praktikfallet

Ni ska planera och styra följande fiktiva projekt:
Projektet avser utveckling av en webbtjänst av det större slaget. Tjänsten erbjuder sökning och bokning av många olika typer av produkter i resebranschen. Följande gäller:

- Ungefärlig uppskattning av storlek på projektet är 10 000 mantimmar och det beräknas löpa över ca 6 månader.
- Full bemanning är ca 15-20 personer, projektledare, delprojektledare, projektadministratör, CM, kravansvarig, användbarhetskoordinator/gränssnittsansvarig, designers, programmerare, testledare, testare.
- Arbetet ska bedrivas iterativt i nära samarbete med representanter från beställaren samt med möjlighet att arbeta direkt med slutanvändare (mannen på gatan) för t ex utvärderingar.

Arbetet omfattar bl a

- alla aktiviteter från ax till limpa, dvs från inledande verksamhets- och behovsanalyser till den slutliga lanseringen av tjänsten
- utveckling av affärslogik och koppling till existerande system
- utformning och konstruktion av interaktionsgränssnittet för tjänsten

Frågor till stöd för diskussionen

- Hur ser processen ut - hur, vad och när itererar ni?
- Vilka leveranser kan ni lova till beställaren, innehåll och tidpunkter för leveranserna? När och hur kan ni beskriva dessa leveranser?
- Vilka avstämningspunkter krävs - när, med vem och om vad?
- Hur vet ni att ni är klara?
- Hur följer ni upp projektet?
- Hur ser projektledningens arbete ut?
- Vilka verktyg och vilket stöd är nödvändigt för att klara av arbetet med att planera och styra projektet?

Resultat

Sammanfattning

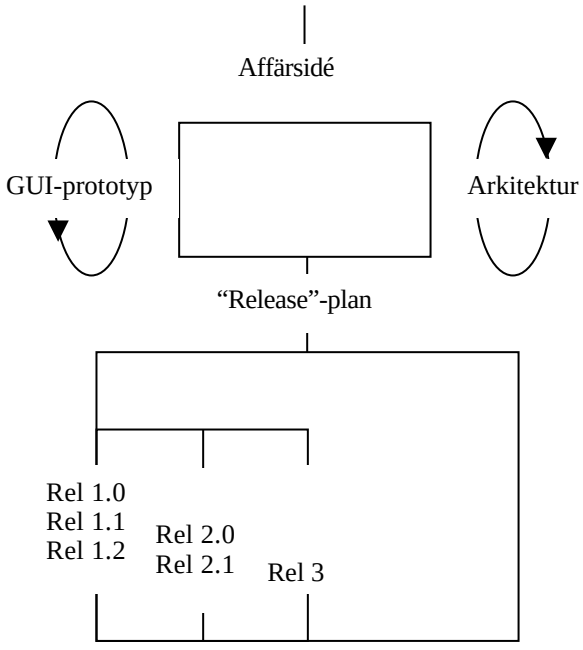
I stort sett alla grupper presenterade sitt resultat i form av en modell för processen. Mycket längre än så hann man inte i sina diskussioner. Trots relativt snäva förutsättningar, där den iterativa processen var en, skiljde sig resultaten åt i mångt och mycket. Vissa såg iterationer som att dela upp systemet i olika delar som levereras vid olika tidpunkter, andra delade upp processen i “fronter” baserade på användaren, utvecklaren och beställaren, där varje “front” motsvarade vissa aktiviteter.

Diskussionerna inte bara svarade på frågor utan skapade även nya. Som exempel diskuterades prototyper och synen på dem – är den i princip en “mall” för den slutliga produkten, eller är den diskussionsunderlag där föränderligheten är en positiv egenskap.

Några av de processer grupperna skissade beskrivs nedan.

Dela upp i “releaser”

En av processerna byggde på ett “release”-förfarande – där varje “release” eller leverans sattes i drift hos användarna. Processen startar med affärsidé, mål och vision. Parallellt med framtagandet av en prototyp av användargränssnittet, skissar man på systemarkitekturen. Med hjälp av prototypen tar man fram övergripande krav. Sedan skapar man en “release”-plan, där varje del omfattar detaljerad kravanalys, implementering och acceptanstest. De olika leveranserna omfattar dels en specifik del av helheten, dels en iteration av de tidigare levererade delarna. Varje leverans innebär att en viss del av systemet sätts i drift hos användarna. På så sätt kan man återföra kunskap till projektet från den verkliga användningssituationen. Avstämningspunkter läggs in efter varje leverans.Figuren nedan illustrerar processen



Arbeta på tre fronter

I en av processerna delades arbetet upp på olika fronter. Processen inleddes med framtagning av en vision tillsammans med beställaren. Sedan identifieras målgruppen enligt beställarens åsikt. Efter detta gör projektet en egen “feasability study” där man undersöker tekniska samt ekonomiska förutsättningar för projektet. Alla dessa aktiviteter itereras.

Sedan infaller en beslutspunkt – som i förekommande fall resulterar i ett avtal eller motsvarande.

Därefter delar man in processen i nedanstående “fronter” – arbetet bedrivs iterativt även i fortsättningen, med beslutspunkter efter varje iteration.

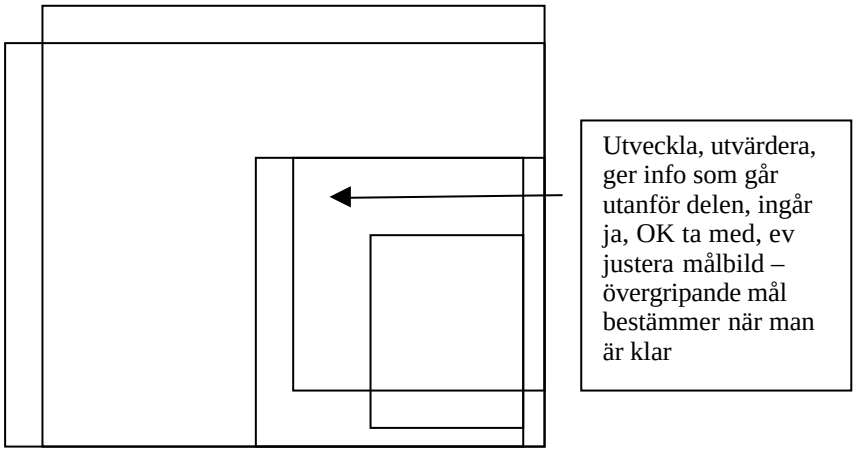
- Användbarhet – användare – den kunskap som växer fram här genererar kraven.
- Systemdesign – utvecklarna – inledningsvis omfattar arbetet användningsfall, prototyper och tester av dessa – resultatet här är kunskap inom de andra områdena.
- Kravdefinition – beställaren – kraven formas utgående från de andra fronterna. De genererar en systemdesign.

Det finns två sätt att se på processen och på frågan när man är klar.

- Iterationerna under den andra delen av processen (med de tre fronterna) ger en successiv utveckling av produkten
- Iterationerna under den andra delen är bara ett redskap för att förfina kravspecifikationen. Denna går sedan vidare till implementation.

Lapptäcksteknik

En av processerna byggde på en “lapptäcksteknik”, se figuren nedan.



"Vi och efter vi"

Systemets/produktens hela liv kan ses som en cirkel med en genomskärning där den ena halvcirkeln representerade utvecklingsprojektet, den andra användningen. Utvecklingsprojektet arbetar fram till lanseringsdatum och en tid efter, men sedan lever systemet vidare. Om utvecklingsprojektet gjort ett tillräckligt bra arbete kommer beställaren kanske tillbaka efter ett tag och vill ha mer. Vissa krav finns för när systemet eller produkten ska lanseras, men klar blir den aldrig.

I den här cirkeln finns punkter där en "coordination crew" koordinerar och vad som ska göras med nästa prototyp. De koordinerar också hela projektet.

Prototyper

En av grupperna ställde frågan hur vi ska se på prototyper. Är prototypen ett diskussionsunderlag (mjuk, föränderlig, "fuzzy") eller ska den vara så lik slutprodukten som möjligt så att man får högsta möjliga precision i testerna?

Om man jämför med industridesign, används ordet prototyp på ett mycket vagt och luddigt sätt i IT-branschen. Inom industridesign är prototypen en mall för produkten, och måste därmed vara exakt lik denna. Man använder andra namn för de gestaltningstekniker som används tidigare i designprocessen. Inom mjukvaruutveckling tenderar vi att kalla allt från snabba skisser till datorbaserade fullt fungerande system för prototyper. Vi behöver ett mer exakt språk för att diskutera de tekniker vi använder i designprocessen. Att använda ordet prototyp kan skapa felaktiga associationer och allt för höga förväntningar. Det är viktigt att definiera hur lik prototypen ska vara slutprodukten, hur man ska testa den, vad den ska representera, etc.

Slutsatser och fortsättning

Syftet med dagen var att diskutera iterativ utveckling – vad är det, hur genomför man praktiskt ett iterativt utvecklingsprojekt, hur arbetar man iterativt över hela systemets eller produktens livslängd. Debatten gav inte så mycket konkreta svar som nya frågor.

Några slutsatser som kan dras från dagens diskussioner är dock

- Vi behöver mer kunskap om hur iterativ utveckling fungerar i praktiken. Den affärsmässiga sidan av systemutveckling pekades ut som ett hinder för iterativ utveckling.

- Yrkesskicklighet och erfarenheter är viktigare än modeller för att man ska lyckas inom ett projekt.

- Modeller för systemutvecklingen behövs ändå – det är viktigt med en gemensam plattform i ett projekt. Vad som är mindre viktigt är vilken modell man väljer. Färdiga, kommersiella modeller har den fördelen att de åtminstone försöker erbjuda lösningar till en hel del av de problem som finns inom systemutvecklingen, dvs ett antal personer har satt sig ner och tänkt igenom problemen och möjliga lösningar.

En naturlig slutsats blir, enligt vår mening, att diskussionen borde fokusera mindre på modellen A kontra modellen B och mer på följande frågor:

- Iterativ utveckling – hur fungerar det i praktiken?

- Hur kan vi anpassa den affärsmässiga sidan av ett utvecklingsprojekt till den osäkerhet som finns inbyggd i den iterativa utvecklingsmodellen? Kan vi arbeta oss bort från den traditionella beställar-/leverantörs-situationen och få till stånd en utvecklingsprocess i vilken användare och utvecklar samverkar? Vi måste inse att systemutveckling innebär att man “skjuter på levande mål”, kraven förändras under utvecklingens gång.

- Hur för vi yrkesskicklighet och kunskap baserade på erfarenheter vidare på bästa sätt? Behöver vi definiera specifika utbildningsprogram för att ta hand om dessa saker eller handlar det om att förse de som redan arbetar i verksamheten med bättre kunskaper och möjligen bättre fungerande verktyg?

Deltagare

Andersson, Ylva	Siemens-Elema AB
Arvidsson, Gunhild	Sockholms Universitet
Berglund, Alf	TietoEnator AB
Boivie, Inger	TietoEnator AB/CID
Bonnert, Anna	Ericsson Radio Systems
Broberg, Ulf	Ericsson Radio Systems
Challis, Ted	Botkyrka kommun
Dahlgren, Inger	TietoEnator AB
Degnes Kjaerheim, Torun	Siemens-Elema AB
Dittrich, Yvonne	Universitet Ronneby/Karlskrona
Eneroth, Harald	Stockholms Universitet, DSV
Englund, Annica	Icon Medialab
Eriksén, Sara	Universitet Ronneby/Karlskrona
Fatton, Ann	Kungliga Tekniska Högskolan, IPLab
Fredriksson, Mats	TietoEnator AB
Gulliksen, Jan	Uppsala Universitet/CID
Gustafsson, Nils-Erik	Ericsson Utveckling AB
Göransson, Bengt	Enea Redina
Hagström, Anders	Ericsson Mobile Communications
Hellqvist, Maja	Tech Lead
Lantz, Ann	CID, Kungliga Tekniska Högskolan
Larsson, Pär	Ericsson Mobile Communications
Levander, Kerstin	Cap Gemini
Lindquist, Sinna	CID, Kungliga Tekniska Högskolan
Lundberg, Christina	Siemens-Elema AB

Lysholm, Frida	Siemens-Elema AB
Osborne, Trevor	TietoEnator AB
Osmund, Gunilla	Enea Redina
Palmqvist, Kristian	Skandia Liv
Pargman, Daniel	Linköpings Universitet
Pastuhov, Rami	Skandia Liv
Petrov, Peter	CID, Kungliga Tekniska Högskolan
Philipson, Thomas	Tech Lead
Rodrigues, Henrry	Kungliga Tekniska Högskolan
Sandquist, Ingeborg	Cap Gemini
Skoogh, Per-Magnus	DSDM-konsortiet/Open World Managment
Stjernqvist, Inger	Kungliga Tekniska Högskolan, Grafisk Teknik
Swerup, Jan	Ericsson Mobile Communications
Söderholm, Eva	Siemens-Elema AB
Öbrink, Henrik	Init
Öhlén, Eva	Cap Gemini
Öman, Björn	Ericsson Mobile Communications