LUNDS UNIVERSITET

Institutionen för informatik

Wikiverktyg inom systemutveckling

En studie om kunskapshantering inom systemutveckling med hjälp av wikiverktyg

Kandidatuppsats, 15 högskolepoäng, SYSK01, informatik

Framlagd: 2011-12-21

Författare: Pontus Byqvist

Anatoly Mironov

Handledare: Magnus Wärja

Examinatorer: Andreas Jacobsson

Claus Persson

Titel: Wikiverktyg inom systemutveckling: En studie om kunskapshantering inom systemutveckling med hjälp av wikiverktyg

Författare: Pontus Byqvist, Anatoly Mironov

Utgivare: Institutionen för informatik

Handledare: Magnus Wärja

Examinatorer: Andreas Jacobsson, Claus Persson

Publiceringsår: 2011

Uppsatstyp: Kandidatuppsats

Språk: Svenska

Nyckelord: Wiki, wikiverktyg, kunskapshantering, kunskapshanteringssystem, systemutveckling, dokumentation, kollaboration, webb 2.0

Abstrakt

I denna uppsats har fokus legat på tre olika begrepp: kunskap, wikiverktyg och systemutveckling. Kunskap och kunskapshantering är begrepp som har funnits i många år och är i sig inga nya fenomen. Vad som dock är nytt är hur kunskapen kan hanteras med just olika webbverktyg, där wiki är ett av dessa verktyg. En wiki är idag ett välkänt verktyg för att dokumentera kunskap och kollaborera via ett webbgränssnitt. En wikis främsta egenskaper är att den är dynamisk, att den är lätt att redigera i och att den främjar jämställdhet. Det tredje uttrycket, systemutveckling, är det som gör att denna uppsats sticker ut från de andra uppsatserna som har gjorts om wikiverktyg. Vårt syfte var alltså att identifiera på vilket sätt wikiverktyg används inom systemutvecklingsorganisationer för att hantera kunskap, vilket även låg till grund för vår forskningsfråga: *Hur används wikiverktyg inom systemutveckling för att hantera kunskap?*. Sättet vi angrep denna forskningsfråga på var genom att genomföra semi-strukturerade intervjuer på fyra olika organisationer. Frågorna vi ställde var direkt kopplade till vår forskningsfråga och de fokusområden vi hade tagit fram i vår undersökningsmodell. Vårt slutresultat visade på bland annat att valet av systemutvecklingsmetod inte påverkade viljan att dokumentera inom ett projekt, att wikiverktyg ofta används på ett projektöverskridande sätt, att en wiki underlättar vid dokumenterandet inom systemutvecklingsprojekt och att det finns situationer där andra verktyg lämpar sig bättre för kollaboration och dokumentation än en wiki.

Innehållsförteckning

1. Introduktion....1
   1.1. Problemformulering och syfte....1
   1.2. Avgränsningar....2
2. Litteraturgenomgång....3
   2.1. Kunskap....3
      2.1.1. Skillnaden mellan kunskap, information och data....3
      2.1.2. Skillnaden mellan outtalad och explicit kunskap....4
      2.1.3. Kunskapskonverteringslägen....5
      2.1.4. Nonakas kunskapsspiral....6
      2.1.5. Fyra kunskapsaktiviteter....7
      2.1.6. Olika former av kunskapsskapande....8
         2.1.6.1. Ba och socialt kapital....8
         2.1.6.2. Nonakas teori om kunskapsskapande....9
   2.2. Wikiverktyg....9
      2.2.1. Vad innebär webb 2.0?....9
      2.2.2. Vad är en wiki?....10
         2.2.2.1. Vad är en enterprise wiki?....11
      2.2.3. Kunskapshantering med hjälp av wikiverktyg....12
         2.2.3.1. Kunskapskonvertering med hjälp av wikiverktyg....13
         2.2.3.2. En wikis egenskaper....14
         2.2.3.3. En wikis designprinciper och påverkan på kunskapshantering....15
         2.2.3.4. Wikianvändning i allmänhet....17
         2.2.3.5. Wikianvändning vid systemutveckling....17
   2.3. Systemutveckling....18
      2.3.1. Vattenfallsmodellen....18
      2.3.2. Agila metoder....19
      2.3.3. Low-technology och gripbarhet....21
         2.3.3.1. Vad är kanban och scrumboard?....22
   2.4. Undersökningsmodell
  23
      2.4.1. Kunskap....24
      2.4.2. Wikiverktyg....24
      2.4.3. Systemutveckling....25
3. Metod och empirisk undersökning....27
   3.1. Semi-strukturerade intervjuer och urval....27
   3.2. Intervjuupplägg....28
   3.3. Datainsamling och analysmetod....30
   3.4. Kvalitet (reliabilitet, validitet och etik)....31
4. Empiri och resultat....33
   4.1. Undersökningsobjekt....33
   4.2. Empiriska resultat....33
      4.2.1. Kunskap....34
      4.2.2. Wikiverktyg....34
      4.2.3. Systemutveckling....36
5. Analys och diskussion....42
   5.1. Kunskap....42
   5.2. Wikiverktyg....45
   5.3. Systemutveckling....46
6. Slutsatser....49

Bilaga 1 - Ordförklaringar....50
Bilaga 2 - Intervjufrågor....51
Bilaga 3 - Intervjuer....53
Intervju med Deniz på Tretton37....53
Kompletterande intervju med Tobias på Tretton37....57
Intervju med Johan på Syncron....60
Intervju med Hampus på Bool....67
Intervju med Mikael på Haldex....71
Bilaga 4 - RFC-dokument (Request for Change)....77
Bilaga 5 - Organisationsinformation....80
Tretton37 AB....80
Syncron AB....80
Bool Nordic AB....80
Haldex AB....80
Referenser....82

Tabellförteckning

Tabell 2.1: Författares särskiljning mellan data, information och kunskap:....4
Tabell 2.2: Uttrycken av Newman & Conrad gentemot Nonaka:....8
Tabell 2.3: Positiva och negativa egenskaper med en wiki:....15
Tabell 2.4: Designprinciper för en wiki. Tagen ur Müller et al. (2008):....16
Tabell 2.5: En jämförelse mellan Nielsens och Chau & Maurers undersökningar vad det gäller wikianvändning:....18
Tabell 3.1: Tidsestimat för våra intervjuer:....29
Tabell 4.1: Intervjuresultat: Hur används wikiverktyg inom systemutveckling för att hantera kunskap?....40
Tabell 6.1: Intervjufrågorna till våra intervjuer:....51

Figurförteckning

Figur 2.1: Kunskapskonverteringens fyra ursprungliga lägen. Tagen ur Nonaka (1994)....5
Figur 2.2: Nonakas kunskapsspiral. Tagen ur Nonaka (1994)....7
Figur 2.3: Skärmdump från en typisk wiki. Tagen ur Nordin & Öhlin (2009)....10
Figur 2.4: Skärmdump från Confluence Enterprise Wiki, här integrerat med ärendehanteringssystemet Jira. Tagen från Atlassian.com (2011-04-08). 12
Figur 2.5: 90-9-1-regelns pyramid. Tagen ur Lytras et al. (2009)....17
Figur 2.6: Vattenfallsmodellen med iterationer. Tagen ur Ruparelia (2010)....19
Figur 2.7: Jämförelse mellan de agila metodernas abstraktionsnivåer. Tagen ur Moratilla Temprado & Ruz Bendito (2010)....21
Figur 2.8: En typisk kanban. Tagen från Crisp.se (2011-03-30)....22
Figur 6.1: Tretton37:s kärninstanser. Tagen från Tretton37.com (2011-04-13)....54

1. Introduktion

Idag förändras allting i en snabb takt. När människor, teknologi och produkter förändras med tiden är det bara kunskap som återstår (Davenport & Prusak, 1998), då kunskap är tävlingsgrunden i ett postkapitalistiskt samhälle (Drucker, 1991). Ett känt citat för att illustrera detta är taget från IT-organisationen HP:s VD Lewis Platt: ”If only HP knew what HP knows”, som syftar på att kunskap som finns i organisationer inte automatiskt blir organisationens kunskap som effektivt kan användas för bättre affärer (Sieloff, 1999). En organisations kunskap kan hanteras på många olika sätt med flera verktyg, där en wiki är ett sådant verktyg.

En wiki är ett verktyg för att kunna samarbeta via ett webbgränssnitt. Wiki är ett av de främsta verktygen som ingår i kategorin webb 2.0, där användare själva snabbt och enkelt kan redigera innehållet (mer om webb 2.0 i kapitel 2.2.1). Wiki är generellt sett ett relativt nytt koncept, som fick sitt stora genombrott i och med att Wikipedia lanserades 2001, och wiki har idag växt till att bli ett välkänt fenomen.

Wikiverktyg kan ses som ett hjälpmedel till ett problem som många organisationer har idag: e-mail overflow, vilket innebär att det finns ett överflöde av e-post som distribueras inom organisationen (Jenefeldt, 2010). En annan aspekt som wiki främjar är samarbete inom organisationen med fokus på kunskapshantering och dokumentation. Samarbete inom dessa områden leder till rikare innehåll, lättare informationsdelning, ifrågasättande av kunskapen, fler överväganden och bättre förståelse för de tekniska begränsningarna (Burton, 2005).

Organisationer har förändrats enormt i och med webb 2.0. Sådana organisationer som har anammat webb 2.0-teknologin kallas för enterprise 2.0 (Olsson, 2008). Till de stora fördelarna med wikiverktyg hör ökad demokratisering inom organisationer, att medarbetare får snabbare svar på sina frågor, att dolda talanger lättare kan upptäckas och framförallt att all kunskap finns på ett och samma ställe (Pettersson, 2007). Nackdelarna med wikiverktyg är behovet att minnas vad sidnamn för att till exempel kunna länka till dem, att versionshantering över flera wikisidor är komplicerat och att det är svårt att exportera innehållet i wikin till andra verktyg (Decker et al., 2007).

Denna uppsats handlar om wikiverktyg, som är ett etablerat verktyg för att behålla kunskap. Uppsatsen riktar sig främst till personer som arbetar med systemutveckling, då det är inom denna yrkesgrupp som den empiriska insamlingen har skett.

1.1. Problemformulering och syfte

Wikiverktyg är ett etablerat verktyg för kunskapshantering i organisationer (Jenefeldt, 2010). Andersson & Rollof (2010) har undersökt hur wikiverktyg används ute på olika typer av organisationer idag. Den här uppsatsen tar fasta på Andersson & Rollofs arbete, men utvecklar studien genom att främst undersöka systemutvecklingsorganisationer. Syftet med denna uppsats är att identifiera på vilket sätt wikiverktyg används inom systemutvecklingsorganisationer för att hantera kunskap.

Uppsatsens forskningsfråga är därför följande:

Hur används wikiverktyg inom systemutveckling för att hantera kunskap?

Då kunskapshantering är ett brett ämne valde vi att främst studera hur kunskap dokumenteras, men vi har även tagit hänsyn till hur kunskap fångas upp och sprids med hjälp av en wiki. För att kunna besvara forskningsfrågan har tre fokusområden tagits fram och dessa har sedan legat till grund för hela uppsatsen. Fokusområdena är som följer:

- Kunskap
- Wikiverktyg
- Systemutveckling

1.2. Avgränsningar

För att begränsa området och göra uppsatsen hanterlig har vissa avgränsningar gjorts. Det kan vara intressant att undersöka vad som motiverar människor att bidra till att skriva ner sin kunskap på en wiki. Vi som författare har dock bestämt att detta inte faller inom forskningsområdet och detta, tillsammans med möjliga incitament, kommer inte undersökas i uppsatsen.

För att kunna implementera en wiki på en organisation krävs vissa tekniska kunskaper. Detta är ännu ett ämne som inte faller innanför ramarna för forskningsfrågan och därför beskrivs inga tekniska detaljer för själva implementeringen av en wiki i uppsatsen.

Vi kommer inte heller att undersöka hur kunskapsspridningen inom en organisation förändras eller hur användandet av en wiki påverkar kompetensen eftersom det finns flera faktorer som kan samspela än bara själva användandet av ett kunskapshanteringssystem.

2. Litteraturgenomgång

I detta kapitel beskrivs de vetenskapliga teorier som använts för att kunna beskriva de metoder och det empiriska tillvägagångssätt som brukats i uppsatsen. Teorierna kommer i den mån möjligt att ställas mot varandra för att på så sätt skapa en mer överskådlig bild över vad det finns för likheter och skillnader dem emellan. Kapitlet avslutas med att det teoretiska ramverket presenteras.

2.1. Kunskap

Idag lever vi i ett kunskapssamhälle (Hargreaves, 2002). Kunskap är centralt på en marknad där organisationer konkurrerar med idéer och innovation. I detta kapitel samlas teorier om kunskap som är relevanta för uppsatsens problemområde. Här presenteras olika definitioner av kunskap, viktiga koncept inom kunskapshantering i organisationer samt kunskapskontext och dess relation med andra begrepp. Först och främst förklaras det vad kunskap har för förhållande till begreppen information och data.

Senare i kapitlet kommer wikiverktygens förhållande till kunskap att beskrivas. En wiki är till för att dela, kombinera och förädla kunskap och genom att använda wikin på ett sätt som stödjer kunskapshantering kan wikin klassificeras som ett kunskapshanteringssystem. (Andersson & Rollof, 2010)

2.1.1. Skillnaden mellan kunskap, information och data

Kunskap och information är två begrepp som ofta behandlas som om de vore utbytbara. Det finns dock en tydlig skillnad mellan dessa två begrepp. Information är alla meddelanden som produceras. Information är en förutsättning för att kunskap ska kunna finnas, då kunskap använder, rättar till och anpassar informationen. Denna kunskap kan sedan en agent, den personen som innehar kunskapen, göra till sin egen personliga kunskap, vilket sker genom ett personligt åtagande. (Nonaka, 1994)

En avgörande egenskap för att skilja information och kunskap åt är var de finns. Kunskap finns i ett ba, vilket är en plats med delade erfarenheter och idéer (mer om ba i kapitel 2.1.6.1). Informationen däremot finns i media och nätverk. Information är gripbart medan kunskap är ogripbart. (Nonaka & Konno, 1998)

Data är objektiv fakta om något som händer. Data besvarar frågor som vad, hur mycket och när. Data avslöjar inte varför något hände eller om det kommer att hända igen. När den begåvas med syfte och betydelse blir data till information. (Davenport & Prusak, 1998)

Data är starkt förknippat med IT-system, då den går att lagra i organisationsdatabaser och är grunden för att kunna göra analyser och se samband mellan händelser. Data beskrivs också som icke bearbetad, alltså "oprocesserad".

Information uppstår av data som har processerats. (Machlup, 1980, refererad i Alavi & Leidner, 2001)

Kunskap är det som en agent, alltså den personen som innehar kunskapen, känner till. Det kan inte finnas kunskap utan att någon känner till den. Även om kunskap kan skrivas ner och bäddas in i organisatoriska processer, nätverk och rutiner kan den inte kallas för kunskap förrän det finns inuti huvudet på en agent. Kunskap uppstår och existerar i sin sanna form enbart inuti människors huvuden. (Fahey & Prusak, 1998) Machlup (1980, refererad i Alavi & Leidner, 2001) använder ordet "autentiserad" för att beskriva informationen som en individ har gjort till sin egen medan Nonaka (1994) talar om kunskap som "aspirerad information".

Kunskapshantering är något som alla organisationer måste ta stor hänsyn till, då det krävs för att organisationen ska överleva. Att bara ha ett kunskapshanteringssystem garanterar inte att det används utan kulturen på arbetsplatsen avgör hur kunskapen delas och skapas. Studier har påvisat att i organisationer där medarbetare har delat med sig och skapat gemensam kunskap har systemet tagits emot med entusiasm och en allmän positiv känsla. Organisationer där kunskapsdelningen har varit minimal har misslyckats med implementeringen av kunskapshanteringssystemet. (Vandenbosch & Ginzberg, 1997, refererad i Alavi & Leidner, 1999)

Tabell 2.1: Författares särskiljning mellan data, information och kunskap:

	Data	**Information**	**Kunskap**
Davenport & Prusak	Objektiv fakta	Data begåvat med syfte	Det som är inbäddat i huvudet på en agent
Fahey & Prusak	Skapas vid införandet av ny stimuli	Skapas vid införandet av ny stimuli	Det som är inbäddat i huvudet på en agent
Machlup	Oprocesserad	Data som är processerad	Autentiserad information
Nonaka & Konno	Explicit kunskap	Alla meddelanden som produceras (gripbart)	Anpassning av informationen, aspirerande information (ogripbart)

2.1.2. Skillnaden mellan outtalad och explicit kunskap

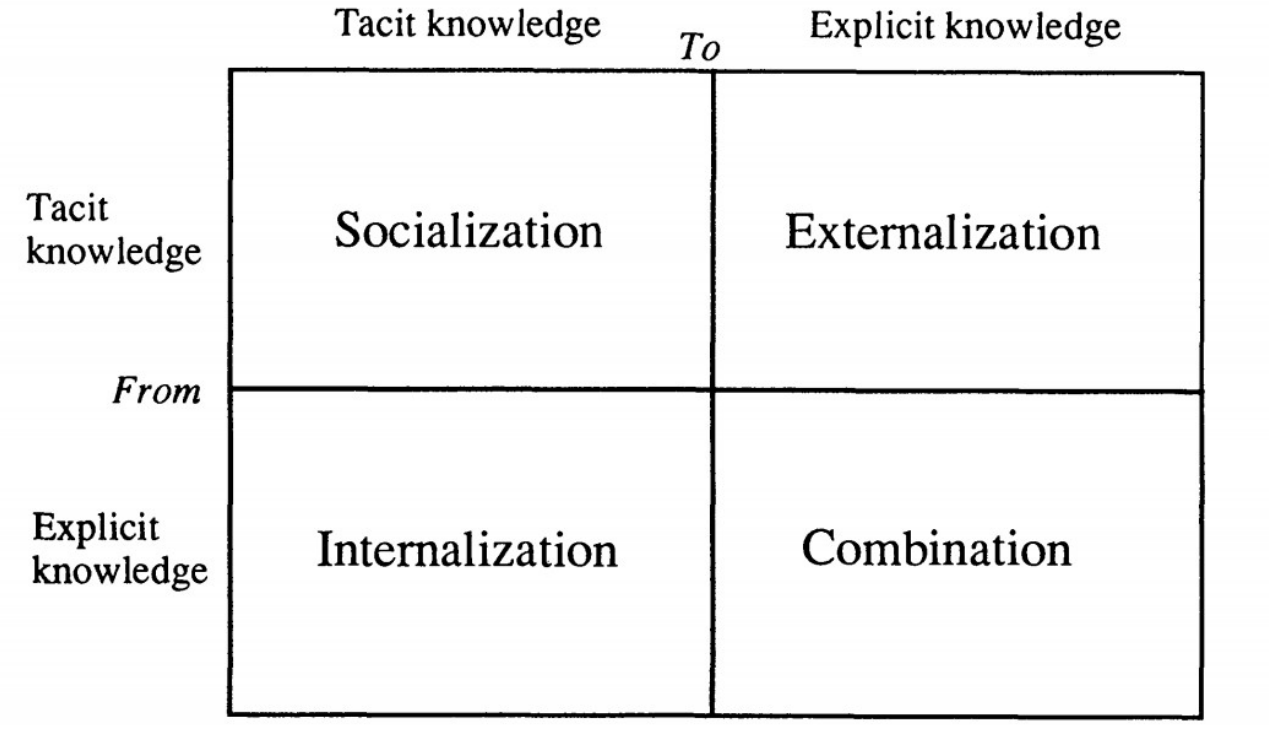
Inte all kunskap som människor besitter går att uttala eller skriva ner. Kunskap som kan uttryckas med siffror och ord är bara toppen av isberget. (Nonaka, 1994) Just den kunskap som blivit formulerad kallas explicit. Motsatsen är outtalad kunskap (tacit knowledge), där denna typ av kunskap även kallas tyst (Andersson & Rollof, 2010) eller implicit (Boden et al., 2009). Den typen av kunskap finns hos en individ eller en grupp som har en "delad upplevelse", vilket är en del av Socialization-läget (Nonaka, 1994), och även om den outtalade kunskapen inte finns formulerad utgör den en grund för handling.

2.1.3. Kunskapskonverteringslägen

Kunskap är ingen oföränderlig sanning, vilket är utgångspunkten i den så kallade epistemologiska synen på kunskap. (Nonaka, 1994) Kunskap utvecklas hela tiden (Fahey & Prusak, 1998). Den ontologiska synen (i motsats till epistemologiska synen) på kunskap, presenterad i Nonaka (1994), förutsätter att kunskap är en dynamisk, mänsklig process av anpassande av personliga föreställningar och övertygelser. Denna dynamiska anpassning av personliga övertygelser refereras till som kunskapskonvertering. (Nonaka, 1994) Eftersom det finns två typer av kunskap, outtalad och explicit, finns det fyra (två upphöjt till två) möjliga konverteringslägen:

1. Outtalad till outtalad: Socialization
2. Explicit till explicit: Combination
3. Outtalad till explicit: Externalization
4. Explicit till outtalad: Internalization

Utgångspunkten i figuren nedan är figurrubriken *From* (från) där en rörelse ska ske åt höger mot figurrubriken *To* (till). I rutan Socialization korsas de båda underrubrikerna Tacit knowledge (outtalad kunskap) med varandra och enligt Nonaka (1994) visar detta på att i detta konverteringsläge konverteras outtalad kunskap till outtalad kunskap. Liknande tillvägagångssätt ska sedan upprepas på de tre övriga konverteringslägena.



Figur 2.1: Kunskapskonverteringens fyra ursprungliga lägen. Tagen ur Nonaka (1994).

Socialization handlar om att lära sig genom "delade upplevelser" och att det görs tillsammans. Mycket av kunskapen sitter i fingrarna och är väldigt svår att formulera i text. En ny medarbetare inom organisationen lär sig då ofta genom att titta på och härma.

I Combination-läget skapas kunskap genom att kombinera befintlig explicit kunskap (nedskrivna dokument och tydliga överenskommelser) och ge mervärde med hjälp av aggregering. (Nonaka, 1994)

Socialization- och Combination-lägena är de konverteringslägena som har fått mest uppmärksamhet genom åren, men i sina rena former har de båda sina begränsningar. Brist på individåtagande och försummande av personlig innebörd kan innebära att ren Combination blir en ytlig tolkning av existerande kunskap.

Å andra sidan, "delad kunskap" som skapats endast i ren Socialization, kan vara begränsad och svår att applicera i områden utanför just den specifika kontext den skapats i. (Nonaka, 1994)

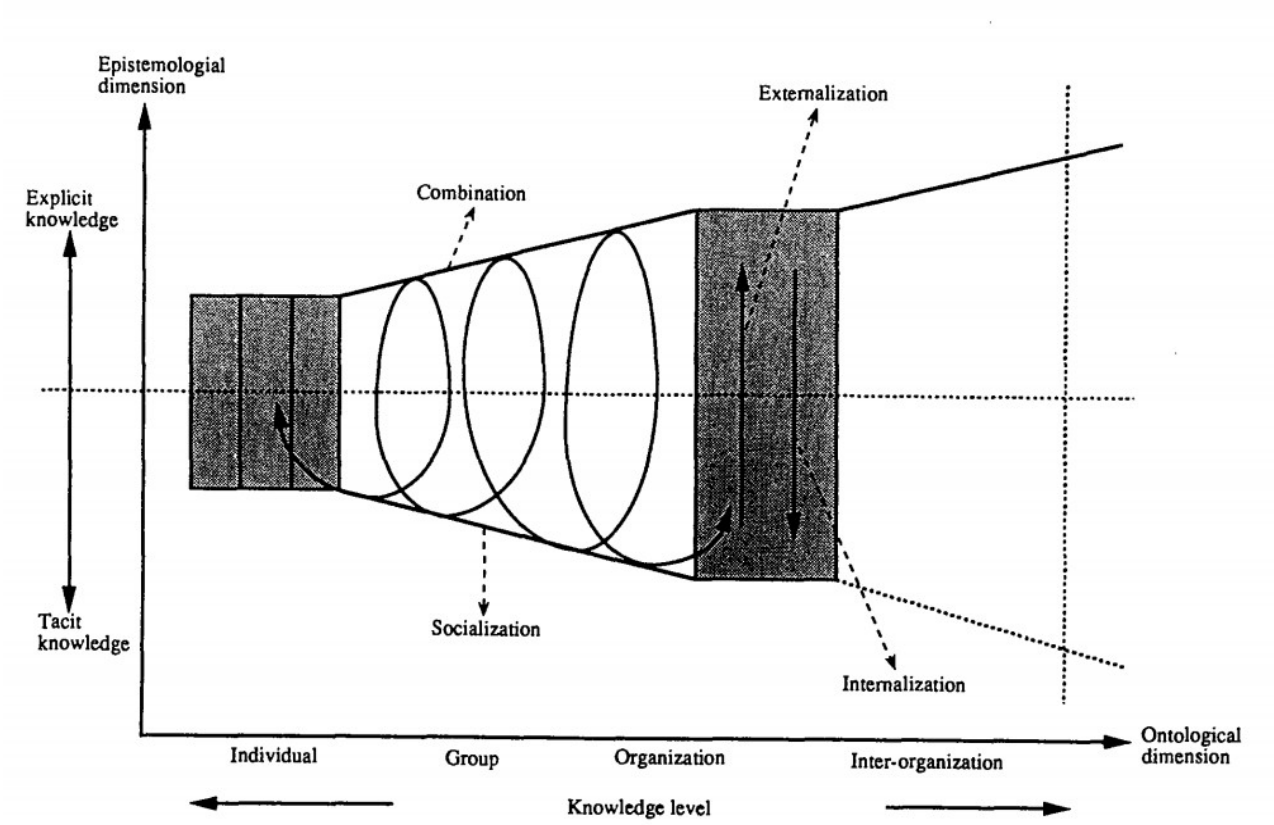
För att Combination-läget ska fungera måste kunskapen även internaliseras. I Internalization-läget omvandlas den explicita kunskapen till outtalad kunskap. Individen tar till sig kunskapen, gör den till sin egen och anpassar den så att den blir ens egna personliga övertygelse. (Nonaka, 1994)

Externalization är det kunskapskonverteringsläge som fått minst uppmärksamhet inom organisationer genom åren (Nonaka, 1994). Omvandlingen från outtalad kunskap till explicit kunskap, som sker i det här läget, är vital för en organisation. Om inte kunskap fångas kommer den att försvinna. Externalization-läget kan därför underlättas med hjälp av metaforer, analogier och modeller. Metaforer är nämligen människors naturliga sätt att tänka och en metafor är ett bildligt uttryck som tar upp gemensamma egenskaper med jämförelseobjektet (som finns i den outtalade kunskapen) och ger en grund för diskussioner om hur kunskap ska bli explicit. (Lakoff & Johnson, 1980, refererad i Nonaka, 1994)

Under senare år har det växt fram ytterligare ett konverteringsläge, utöver de fyra som nämns i figur 2.1. Detta konverteringsläge går under namnet Virtual-Socialization och detta sker via Externalization- och Combination-lägena. Till skillnad från de övriga konverteringslägena så lägger Virtual-Socialization-läget stor vikt vid användandet av diverse IT-verktyg för att underlätta vid kommunikationen mellan individer eller grupper. (Spraggon & Bodolica, 2008).

2.1.4. Nonakas kunskapsspiral

Även om kunskap isolerat kan skapas i var och en av de fyra ovannämnda konverteringslägena sker kunskapsskapandet inom organisationen först när alla fyra lägena kopplas samman till en ständig cykel. Denna kunskapsspiral formas av serier med byten mellan de olika konverteringslägena. Organisationer måste se till att dessa byten fungerar och även uppmuntra de så kallade triggers för bytena mellan konverteringslägena. (Nonaka, 1994)



Figur 2.2: Nonakas kunskapsspiral. Tagen ur Nonaka (1994).

2.1.5. Fyra kunskapsaktiviteter

För att kunna hantera de aktiviteter som har med kunskap att göra har Newman & Conrad (2000) skapat en modell, kallad The General Knowledge Model, som består av fyra steg. Dessa steg är alla aktiviteter och de förklarar hur kunskap skapas och hanteras. (Newman & Conrad, 2000)

Den första aktiviteten består av skapandet av kunskap (knowledge creation). Denna aktivitet går ut på att ny kunskap introduceras i systemet. Aktiviteten inkluderar även kunskapsutveckling, upptäckandet av ny kunskap och infångandet av denna kunskap. Denna aktivitet kan liknas vid Socialization- och Combination-lägena som Nonaka (1994) har beskrivit.

Nästa aktivitet går ut på att bibehålla kunskapen (knowledge retention). Denna aktivitet fokuserar på att bevara den nya kunskapen och den ser även på hur lönsamt det kan vara att behålla informationen inom systemet. Denna aktivitet kan liknas vid Internalization-läget som Nonaka (1994) har beskrivit.

Den tredje aktiviteten fokuserar på att överföra kunskap (knowledge transfer). Här förs kunskapen från systemet över till andra system eller till grupper av människor. Detta sker genom fler olika subaktiviteter såsom kommunikation, tolkning och översättning av information. Denna aktivitet kan liknas vid Externalization-läget som Nonaka (1994) har beskrivit.

Den sista aktiviteten består av användandet av kunskap (knowledge utilization). Här kopplas kunskapen till de affärsprocesser som finns inom organisationen. Det rör sig alltså om praktisk tillämpning av den nyvunna kunskap som erhållits tidigare. Något liknande konverteringsläge finns inte i de teorier som Nonaka har beskrivit.

Tabell 2.2: Uttrycken av Newman & Conrad gentemot Nonaka:

Newman & Conrad	**Nonaka**
Knowledge creation	Socialization och Combination
Knowledge retention	Internalization
Knowledge transfer	Externalization
Knowledge utilization	_

2.1.6. Olika former av kunskapsskapande

Detta kapitel tar upp hur olika författare uppfattar skapandet av kunskap. Uttryck såsom ba och socialt kapitel förklaras och dessa kopplas samman med tidigare nämnda begrepp och teorier.

2.1.6.1. Ba och socialt kapital

Kunskap skapas dynamiskt inom grupper och sociala band spelar en stor roll inom grupper. Dessa band kallas för socialt kapital (Huysman & Wulf, 2004, refererad i Boden et al., 2009). I det begreppet ingår bland annat ömsesidigt stöd, gemensamt språk, överenskommelser, normer, tron på varandra och känslan av en skyldighet att ta hand om varandra. (Huysman & Wulf, 2004, refererad i Boden et al., 2009)

Ett annat koncept som har sin grund i gemenskap är den japanska ba-filosofin. Ba är ett begrepp som utvecklades av den japanske filosofen Kitaro Nishida och som passar väl för att förklara kunskapsskapande. (Nonaka & Konno, 1998)

Ba är en “plats” (bokstavlig översättning från japanska) som delas med andra. Ba kan därför innebära ett geografiskt ställe såsom en arbetsplats, men det kan även vara delade upplevelser, idéer och ideal. (Nonaka & Konno, 1998) Med andra ord, ba är en gemensam nämnare för en grupp som möjliggör kunskapsskapande.

Konceptet ba stämmer också bra in på Virtual-Socialization som vi nämnt ovan där det med hjälp av IT-verktyg sker en virtuell form av Socialization.

2.1.6.2. Nonakas teori om kunskapsskapande

Nonakas forskning berör ofta kunskap och hur den skapas, men för att förstå hur kunskap skapas måste ett ramverk tas fram. Detta ramverk kopplas sedan till Nonakas teorier om kunskapsskapande och tillsammans ger dessa teorier om hur både kollektiv kunskap och innovativ kunskap kan skapas. (Kimmerle et al., 2010)

Kunskapsskapande är ingen enkel process. Nonakas teori om kunskapsskapande har en viktig beståndsdel i sig för att den ska fungera, nämligen att kunskap måste distribueras för att kunna skapas. Nonakas teori kräver att kunskap delas mellan människor och detta gäller speciellt inom organisationer. (Kimmerle et al., 2010) För att kunna dela med sig av sin kunskap finns det många verktyg. Ett av dessa är just vårt studieobjekt, nämligen wiki, som vi kommer att gå djupare in på vad det egentligen är härnäst.

2.2. Wikiverktyg

I detta kapitel beskrivs begreppet webb 2.0 och varför det är så att en wiki faller in under denna benämning. En förklaring på vad en wiki egentligen är görs också här samt en särskiljning på begreppen wiki och enterprise wiki.

2.2.1. Vad innebär webb 2.0?

Webb 2.0, ibland stavat web 2.0, är ett begrepp som myntades i oktober 2004 av Tim O'Reilly, grundaren av teknikförlaget O'Reilly Media, och MediaLive International. Uttrycket webb 2.0 syftar på att de organisationer som överlevde när IT-bubblan sprack under hösten 2001 hade något gemensamt. Tim O'Reilly och webbpionjären Dale Dougherty ställde sig då frågan om det kunde vara så att denna händelse markerade en förändring för webben för alltid. De kom fram till att så var fallet och denna förändring döpte de då till webb 2.0. (O'Reilly, 2007)

Cormode & Krishnamurthy (2008) förklarar begreppet webb 2.0 på följande sätt:

"'Web 2.0' is a term that is used to denote several different concepts: Web sites based on a particular set of technologies such as AJAX; Web sites which incorporate a strong social component, involving user profiles, friend links; Web sites which encourage user–generated content in the form of text, video, and photo postings along with comments, tags, and ratings; or just Web sites that have gained popularity in recent years and are subject to fevered speculations about valuations and IPO prospects."

Exakt vad som krävs för att en webbplats ska falla under definitionen webb 2.0 är inte fastställt, men det finns vissa riktlinjer för att särskilja webb 2.0-webbplatserna från de "äldre" webb 1.0-webbplatserna.

Tre av aspekterna som brukar skilja de två begreppen åt är teknologin (skript och kod som bygger upp webbplatsen), strukturen (syfte och utformning av webbplatsen) och det sociala (webbplatsens hanterande av grupper och kontakter). (Cormode & Krishnamurthy, 2008)

En annan särskiljning, som är aktuell att göra, är den mellan webb 2.0-webbplatser och sociala nätverk. Webb 2.0-webbplatser är den teknologiska plattform som de sociala nätverken är uppbyggda på och begreppen är därmed inte synonymer med varandra, även om webb 2.0-webbplatser tenderar att ha en struktur som har mer gemensamt med sociala nätverk än med webb 1.0-webbplatser. (Cormode & Krishnamurthy, 2008)

Med möjligheten för alla inblandade att ändra innehåll och se resultatet direkt är wiki-teknologin ett av de främsta webb 2.0-verktygen.

2.2.2. Vad är en wiki?

En wiki är en skriptdriven webbplats som låter besökaren fritt redigera innehållet på denna webbplats (Gonzalez-Reinhart, 2005). Den första wikin, med namnet WikiWikiWeb, skapades av programmeraren Howard G. Cunningham och lanserades den 25 mars 1995. Denna wiki är aktiv än idag, men den största och mest kända wikin i dagsläget är Wikipedia, som är skapad av internetentreprenören Jimmy Wales och som lanserades 2001. (Long, 2006)

Figur 2.3: Skärmdump från en typisk wiki. Tagen ur Nordin & Öhlin (2009).

En wiki visar typiska egenskaper för ett kunskapshanteringssystem med tonvikt på möjligheten att konversera och dessa typer av kunskapshanteringssystem är särskilt effektiva vid skapandet av kunskap inom en organisation.

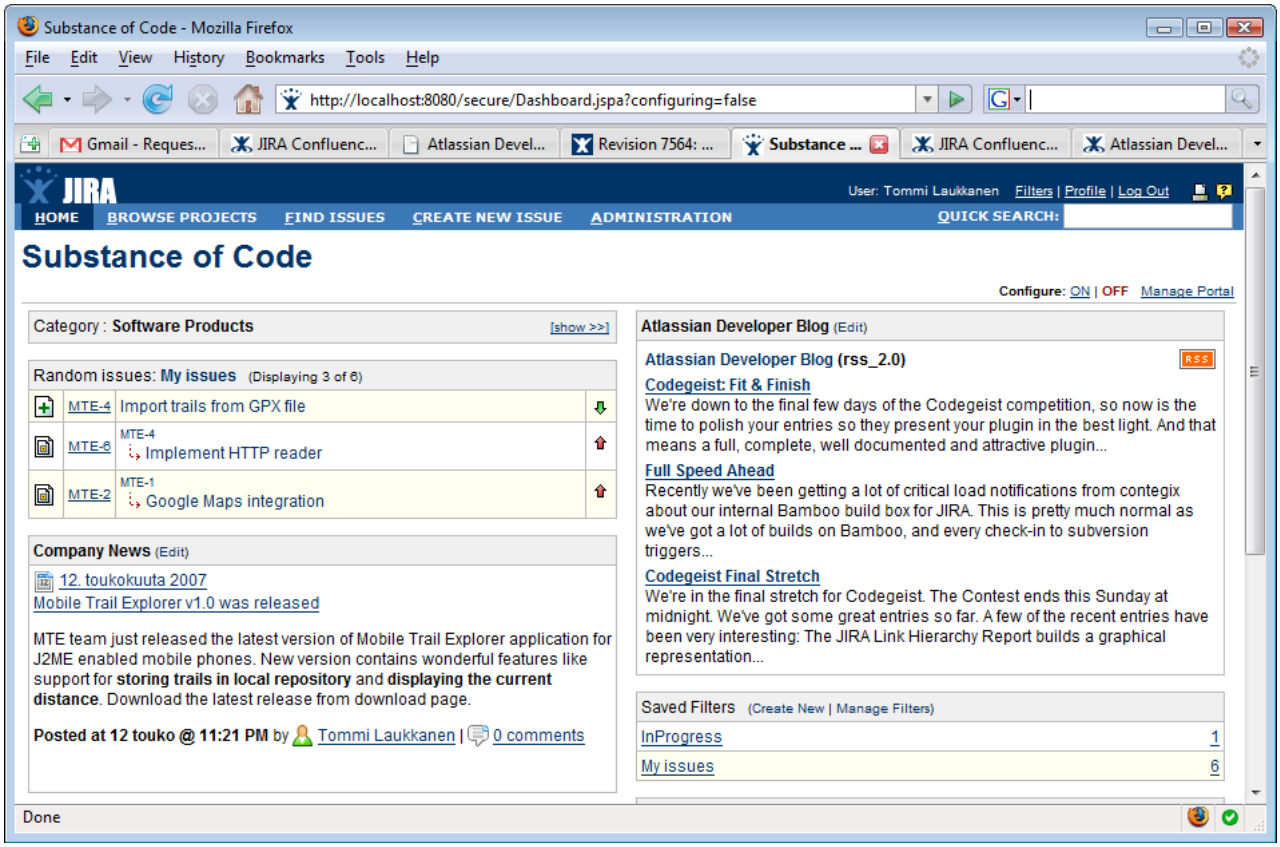
Wikiverktyg har även större potential än andra tekniska konversationssätt inom organisationer idag, såsom e-post, diskussionsforum, snabbmeddelanden och GDSS (Group Decision Support System), vilket är samarbetsverktyg som underlättar vid möten och grupparbeten. (Gonzalez-Reinhart, 2005)

En wiki är generellt sett helt öppen för allmänheten att läsa, redigera och ta bort information från, även om wikiverktyg inom en organisation för det mesta är stängda för att undvika att personer utanför organisationen inte ska komma åt informationen som finns där.

Detta leder till att vissa människor blir osäkra på om det som står i wikin verkligen är sant eller om det är någon illvillig människa som har bidragit med felaktig information på wikin. (Dickerson, 2004, refererad i Gonzalez-Reinhart, 2005) Det finns oftast inga direkta hinder mot att fylla i felaktig information, men i de flesta större wikiverktyg, såsom Wikipedia, finns det användare som har tagit på sig uppgiften att rensa bort felaktig information snabbt och effektivt. (Long, 2006)

2.2.2.1. Vad är en enterprise wiki?

Enterprise wiki, ibland kallad corporate wiki, är en fördjupning inom wiki-sfären och berör främst de sorters wikiverktyg som finns ute hos organisationer. En av de största skillnaderna mellan en wiki och en enterprise wiki är att en enterprise wiki har större möjlighet att begränsa vilka personer som ska få tillgång till att läsa och redigera på den, det vill säga att åtkomsten för användarna begränsas. En enterprise wiki kan även vara tillgänglig på två olika sätt, via internet eller genom mjukvara (Jenefeldt, 2010) och den kan finnas inbäddad i både intranät och extranät (Carr, 2007).



Figur 2.4: Skärmdump från Confluence Enterprise Wiki, här integrerat med ärendehanteringssystemet Jira. Tagen från Atlassian.com (2011-04-08).

För att organisationer ska vara framgångsrika med implementeringen av en enterprise wiki behövs det ta hänsyn till två aspekter. Den första av dessa aspekter är hur enterprise wikin skär ner kostnaderna och hur den främjar försäljningen för organisationen. (Carr, 2007)

Den andra aspekten är hur balansen mellan flexibilitet och kontroll ska hanteras inom organisationen. Det underlättar att vara kontrollerande då det gäller att behålla informationen inom organisationen, men i själva organisationen går det bra att vara flexibel. (Jenefeldt, 2010)

Hädanefter i uppsatsen kommer begreppen wiki och enterprise wiki att användas som synonymer, då respondenterna ofta använde sig av uttrycket wiki för att beteckna båda dessa typer.

2.2.3. Kunskapshantering med hjälp av wikiverktyg

Organisationer idag går från den traditionellt centraliserade och hierarkiska uppbyggnaden till en mer decentraliserad och nätverksbaserad uppbyggnad. Detta leder till att organisationer idag främjar öppenhet på ett annorlunda sätt än vad de har gjort tidigare. På grund av detta ses de anställda och informationen inom organisationen som huvudtillgångar. (Barabási, 2003, refererad i Müller et al., 2008) Platta organisationer, flexibla organisationsstrukturer och ett större beroende av kunskap är något som de flesta organisationer ställs inför idag.

Eftersom organisationer rör sig mot en kunskapsintensiv uppbyggnad ställs det högre krav på de anställda, men sällan är det så att *en* anställd har tillräckligt med kunskap för att lösa tvetydiga och komplexa problem. (Cross et al., 2002)

Det krävs därför ett sätt att hantera den kunskap som finns inom organisationen. Inom flertalet organisationer idag används så kallade Knowledge Management Systems (KMS), vars syfte är att stödja skapandet och hanterandet av kunskap inom organisationer. (Alavi & Leidner, 2001) Vid användandet av ett KMS blir den anställde ofta påtvingad en viss struktur på hur detta KMS ska se ut, vilket inte gäller när det rör sig om wikiverktyg. En wikis struktur är mer dynamisk och växer fram efter organisationers, gruppers eller individers behov och önskemål. En wiki kan även underlätta vid processen att konvertera outtalad kunskap till explicit kunskap, Externalization-läget i figur 2.1, då människor arbetar tillsammans med hjälp av en wiki. Detta kan därmed ses som en förbättring jämfört med vanliga KMS (Mader, 2008, refererad i Andersson, 2010).

I de flesta KMS sorteras liknande sidor under varandra på ett hierarkiskt sätt. Detta kan liknas vid ett bibliotek och kallas för taxonomi. Inom wiki-världen däremot så är hierarkin platt och denna struktur går under namnet folksonomi. Utöver detta finns det även stora möjligheter att föra diskussioner och wiki har setts som ett "modifierat diskussionsforum". (Choate, 2008, refererad i Andersson, 2010)

2.2.3.1. Kunskapskonvertering med hjälp av wikiverktyg

Som tidigare nämnts finns det fyra faser av kunskapskonvertering: Socialization, Combination, Externalization och Internalization (Nonaka, 1994). Vi kommer därför här att se kopplingen mellan dessa konverteringsfaser och vårt främsta studieobjekt: wiki.

Socialization innebär att dela med sig av sin outtalade kunskap till andra. Här spelar wiki, som är ett sorts social medium, en stor roll då sociala medier skapar stora utrymmen där den outtalade kunskapen lätt kan delas mellan användarna, (Chatti et al., 2007) En annan variant av Socialization har tillkommit senare, Virtual-Socialization (Spraggon & Bodolica, 2008), vilket är en process som möjliggörs med hjälp av ett IT-verktyget wiki.

Combination betyder systematisering av koncept i ett kunskapssystem och att detta system sedan använder sig av olika typer av explicit kunskap. Wiki fungerar här som ett mer uppdaterat, informationsrikt och lättsökt system än de tidigare, centraliserade systemen. En wiki gör det även lättare för människor att sprida kunskap utanför organisationsgränserna. (Chatti et al., 2007)

Externalization innebär omvandling av outtalad kunskap till explicita koncept. Här är det främst diskussioner användarna emellan som främjar denna fas. Wikiverktyg ger tillfälle att interagera och på så sätt uppstår där ett gemensamt kunskapsskapande. Det finns även möjlighet för användarna att uttrycka sig i mer än ord, såsom till exempel genom bilder, videor och ljud. (Chatti et al., 2007)

Internalization betyder att explicit kunskap tas och görs om till egen outtalade kunskap. Här är det främst en process av inre reflektion som ligger till grund för denna fas. För att effektivt kunna reflektera över sin egen kunskap måste agenten ha en förståelse för hur han eller hon lär sig nya saker. Detta görs genom att studera mönster och genom att öva på förmågan att skapa mening av orden som står på till exempel en wiki. (Chatti et al., 2007)

2.2.3.2. En wikis egenskaper

En wikis redigeringsprocess anses ofta vara lättare att förstå sig på än en vanlig HTML-struktur, det vill säga det märkspråk som är webbstandard idag för textstrukturering. På grund av detta ses wikin mer som ett passande verktyg vid samarbete än en HTML-sida och wikin kan även ta sig an rollen som ett socialt nätverk. Detta brukar klassas som en av de största fördelarna med en wiki och det hela grundar sig i att den oftast är öppen för alla att redigera. (Jing & Fan, 2008)

En wikis viktigaste egenskap är jämställdhet. Ingen wikianvändare diskrimineras på grund av sin ålder, ras, kön, bostadsmiljö, religion, sociala status eller politiska åsikt. En annan aspekt med wikin är att den ses som en platt och decentraliserad organisation, där användarna inte återfinns i någon sorts hierarki och där de fritt kan kommunicera med andra användare inom wiki-plattformen. Sådana rättigheter finns inte alltid inom alla organisationer, vilket har lett till att en wiki har ansetts följa principen att användaren inte behöver hålla med om det som skrivs på wikin, men att denne ändå försvarar rättigheten att dessa åsikter får finnas. (Jing & Fan, 2008)

En wiki kan även ses som en social mjukvara, där användarna (det vill säga de som bidrar till wikin) känner en behörighet och en viss gruppkänsla med de andra användarna på wikin. Detta skapar sociala band mellan användarna och gör i sin tur att användarna får förståelse för hur de ska kunna realisera sina personliga mål. (Boyd, 2003, refererad i Gonzalez-Reinhart, 2005) Detta är en av anledningarna till varför wikiverktyg integreras med annan mjukvara inom organisationer idag (Gonzalez-Reinhart, 2005).

En wiki har dock inte bara positiva egenskaper. I en artikel från 2007 diskuterar Decker et al. några av de största nackdelarna med en wiki. Den första av dessa är att versionshantering mellan wikisidor (”artiklar”) är komplicerat. Även om wikiverktyg använder sig av permalänkar, länkadresser som är mer bestående än genomsnittet, så finns det fortfarande ett manuellt arbete i att samla åt sig dessa permalänkar för att kunna använda dem vid senare tillfällen. (Gonzalez-Reinhart, 2005)

En annan negativ aspekt som tas upp är hur en användare måste komma ihåg det exakta namnet till en viss wikisida för att kunna länka till den. Detta kan innebära ett större arbete för användaren, då denne ständigt och jämt kan tvingas avbryta sin sidredigering för att söka upp andra wikisidors namn. (Gonzalez-Reinhart, 2005)

En tredje negativ sak som tas upp är att det är komplicerat att exportera en wikisidas innehåll till andra dokument. Det kan då vara svårt att bibehålla länkstrukturen när innehållet är flyttat till ett annat dokument, som i sin tur inte nödvändigtvis stödjer wikilänkar. (Gonzalez-Reinhart, 2005)

Tabell 2.3: Positiva och negativa egenskaper med en wiki:

Positivt	**Negativt**
Enkel att redigera på, stor öppenhet	Versionshantering mellan wikisidor är komplicerat
Jämställd, användare har stora rättigheter	Exakta namn till wikisidor krävs
Sociala band uppstår mellan användarna	Export av innehållet på en wikisida är komplicerat

2.2.3.3. En wikis designprinciper och påverkan på kunskapshantering

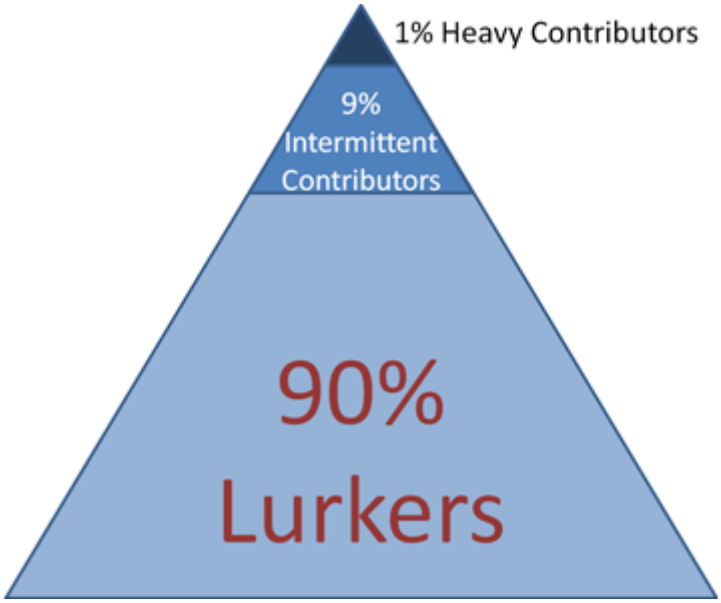
Den första wikin, WikiWikiWeb från 1995, var baserad på ett antal designprinciper (Leuf & Cunningham, 2001, refererad i Müller et al., 2008). Dessa designprinciper har sedan legat som grund för efterföljande wikiverktyg, användarnas beteende runt dessa och utväxlandet av kunskap personer emellan (Müller et al., 2008).

Tabell 2.4: Designprinciper för en wiki. Tagen ur Müller et al. (2008):

Designprincip	**Förklaring**	**Påverkan på kunskapshantering**
Simpel	Det finns få syntaxregler.	Få hinder existerar vid användning och kunskapsdokumentering.
Öppen	Alla användare kan se och redigera innehållet.	Varje användare är kompetent och kunskapen är fri att dela ut.
Inkrementell	Innehållet kan länka till artiklar som ännu inte finns.	Kunskapshålen är synliga och kunskapsutvecklingen är effektiv.
Organisk	Strukturen är öppen för redigering och utveckling.	Kunskap och dess kontext är dynamisk.
Trivial	Vissa textkonventioner har tillgång till användbar kod.	Koden blir mer tillgänglig och därmed lättare att utveckla.
Universell	Principerna är likadana oavsett om man som användare skapar, ändrar eller organiserar innehåll.	Inga definitioner av roller inom kunskapshanteringen är nödvändiga.
Offentlig	För att återskapa ens formaterade output krävs först ens input.	Tillförlitligheten är hög, då det inte är troligt att ens output är påhittad.
Förenad	Sidnamn kräver ingen kontext för att tolka dem.	Artiklarna blir mer lättförståeliga.
Detaljerad	Sidnamn är ordentligt uppbyggda för att undvika förväxlingar.	Kunskapens kontext övervägs.
Tolerant	Oförståeligt beteende förekommer föredragsvis enbart i felmeddelande.	Kodspråk förekommer enbart för de som förstår detta.
Iakttagbar	Sidaktiviteten kan ses över av alla användare.	Kunskapens uppkomst och utveckling analyseras.
Konvergent	Undvik upprepande av information genom att länka till existerande artiklar.	Redundant kunskap slås ihop.
Förtrolig	Det centrala är förtroendet för innehållet.	Framgång beror på organisationens kultur.
Rolig	Det roliga med att lägga till information lyfts fram.	Motivationen för att redigera artiklar är hög.
Delbar	Informationen ska kunna delas ut lätt till alla som vill ha den.	Alla har tillgång till den information de söker, snabbt och enkelt.

2.2.3.4. Wikianvändning i allmänhet

I en undersökning gjord av Nielsen, publicerad i oktober 2006 (Nielsen, 2006, refererad i Lytras et al., 2009), kom han fram till att de flesta användarna som besöker en webb 2.0-sida, alltså en webbplats där användaren själv kan bidra med information, inte bidrar med information själva utan de läser enbart den information som redan finns nedskriven. Ur detta utvecklade Nielsen något som han själv benämner "90-9-1-regeln", vilket betyder att 90 procent av de som besöker hemsidan inte bidrar med någon information själva (dessa kallas lurkers; smygare), 9 procent bidrar med information ibland (dessa kallas intermittent contributors; oregelbundna bidragsgivare) medan den sista procenten bidrar med den största delen av all information (dessa kallas heavy contributors; tungviktsbidragsgivare). Nielsen förklarar att det är omöjligt att få bort denna ojämlikhet, men att det finns sätt att få bukt på detta problem såsom att belöna de som bidrar med information, göra det lättare att bidra med information och lyfta fram de som bidrar med kvalitetsinformation. (Nielsen, 2006, refererad i Lytras et al., 2009)



Figur 2.5: 90-9-1-regelns pyramid. Tagen ur Lytras et al. (2009).

Lytras et al. (2009) utvecklade sedan Nielsens argument och kom fram till att om enbart ett fåtal personer bidrar med information kanske inte systemet är användbart eller blir de som bidrar med informationen överväldigade av allt vad det innebär att bidra med information. Å andra sidan, om det är många personer som bidrar kan det bli svårt att hitta den riktigt användbara informationen bland all information i systemet. Lytras et al. (2009) anser att den bästa lösningen ligger i att ha en handfull välinformerade bidragsgivare, det vill säga en sorts blandning mellan de andra två alternativen.

2.2.3.5. Wikianvändning vid systemutveckling

En undersökning av Chau & Maurer (2005), som utfördes på en medelstor systemutvecklingsorganisation, visade att användarna av den interna wikin på denna organisation till 90 procent bestod av systemutvecklare (de resterande 10 procenten bestod av managers).

Utav de 80 registrerade användarna på wikin var det drygt 60 procent som bidrog med information till wikin. Utav dessa var det 10 stycken användare (12,5 procent av den totala mängden registrerade användare) som stod för 75 procent av informationsbidragen till wikin. Fem av dessa tio stycken stod för 55 procent av informationsbidragen till wikin och de var alla systemutvecklare.

I en annan undersökning av Majchrzak et al. (2006) visade resultatet på att "medelwikin" har 12 stycken användare som bidrar med information och 25 andra användare, som inte på ett väsentligt sätt bidrar med någon information. Undersökningen visar även på att användandet av en enterprise wiki leder till tre positiva effekter. För det första underlättar wikin i användarnas, i detta fall systemutvecklarnas, dagliga arbete. För det andra hjälper wikin organisationen att återanvända kunskap, förbättra processer och att få de anställda att samarbeta. Slutligen förbättrar wikin ryktet inom organisationen för den anställde som bidrar med information.

Tabell 2.5: En jämförelse mellan Nielsens och Chau & Maurers undersökningar vad det gäller wikianvändning:

	I allmänhet (%)	**Inom systemutveckling (%)**
Heavy contributors	1,00	6,25
Intermittent contributors	9,00	12,50
Lurkers	90,00	81,25

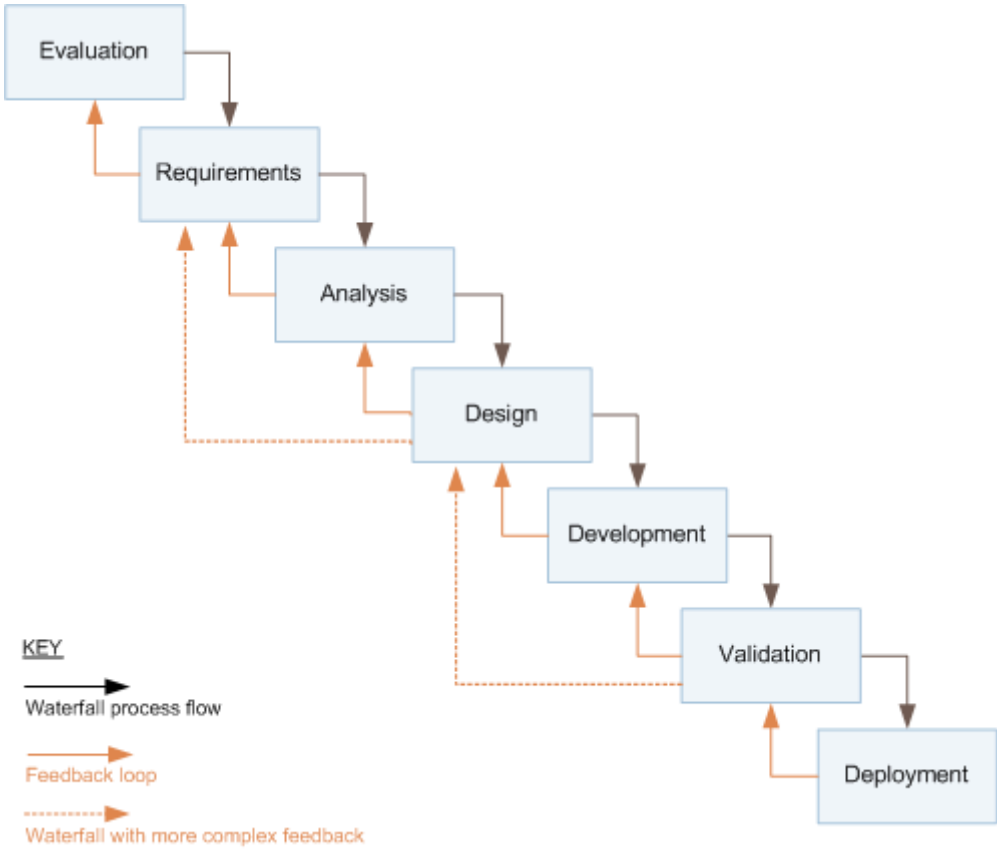
2.3. Systemutveckling

Utöver de tidigare nämnda fokusområdena, kunskap och wikiverktyg, så fördjupar sig denna uppsats även i ämnet systemutveckling. I detta kapitel tas några av de mest använda systemutvecklingsmetoderna idag upp och begrepp såsom agila metoder, low-technology och gripbarhet förklaras.

2.3.1. Vattenfallsmodellen

Vattenfallsmodellen utvecklades ursprungligen av systemutvecklaren Herbert D. Benington år 1956 (Ruparelia, 2010). Enligt denna modell delas systemutvecklingen in i faser, som i figur 2.6. Informatikern Winston W. Royce (1970, refererad i Ruparelia, 2010) förbättrade vattenfallsmodellen med så kallade feedback loops, det vill säga iterationer av förbättringar och återkopplingar, som även dessa går att se i figur 2.6.

Vårt fokus kommer hädanefter att beröra enbart de agila metoderna (som diskuteras härnäst), men vi valde att även nämna vattenfallsmodellen då den fortfarande används inom organisationer idag och för att den fortfarande är en välkänd systemutvecklingsmodell som är värd att känna till.



Figur 2.6: Vattenfallsmodellen med iterationer. Tagen ur Ruparelia (2010).

2.3.2. Agila metoder

Agila metoder är ett paraplybegrepp för arbetssätt inom systemutveckling som fokuserar på en snabbare väg från idé till färdig produkt (Kaplan, 2010). Agila metoder har haft ett enormt inflytande på systemutveckling de senaste åren (Dingsøyr et al., 2010). Även inspirationer från industrier såsom lean och kanban har fått fäste inom systemutvecklingen (Hollingsworth, 2011).

Även om det finns skillnader i synen på kunskapshantering inom olika metoder i agil systemutveckling har de mycket gemensamt i den mån att de flesta av dem fokuserar på social interaktion mellan systemutvecklare i realtid, men även på mindre dokumentation (Ruparelia, 2010). Handling är djupt rotad i den sociala och outtalade kunskapen som inte, eller bara delvis, går att konvertera till explicit kunskap (Polanyi, 1967, refererad i Boden et al., 2009).

Med insikten om att kunskap är kontextuell, vilket innebär att kunskapen är beroende av den kontext den uppstår i, har kunskapshanteringens "andravågsverktyg" börjat användas (Huysman & de Wit, 2004, refererad i Boden et al., 2009).

Till skillnad från klassiska kunskapsdatabaser flyttas här fokus till att stödja informell kunskapsdelning inom grupper med hjälp av olika verktyg (Reichling et al., 2007, refererad i Boden et al., 2009).

Den gemensamma nämnaren för alla agila systemutvecklingsmetoder är snabba leveranser av produkter i mindre iterationer. Det som skiljer dem åt är olika sätt att formulera deras fokus.

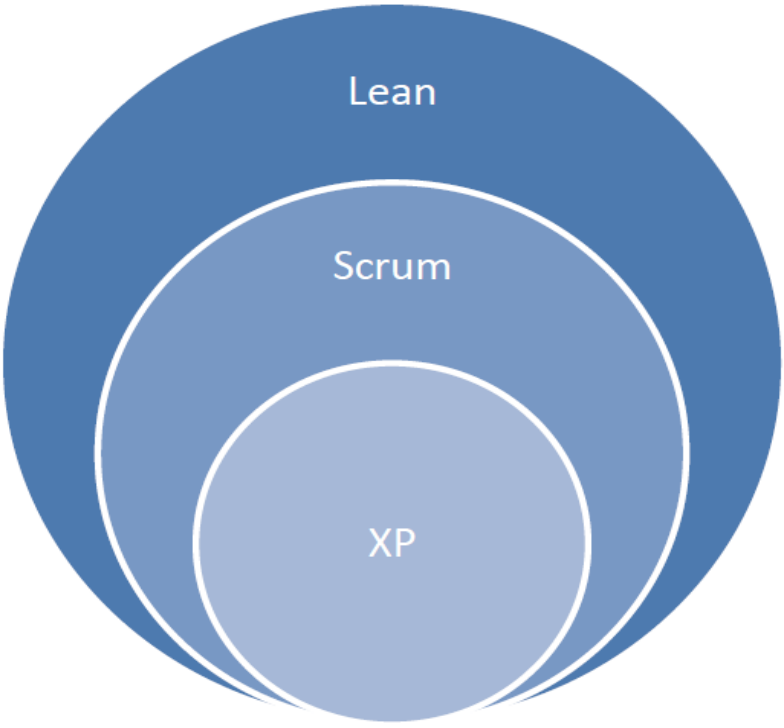
Inom lean software development, eller enbart lean, strävar utövaren åt att eliminera alla moment i systemutvecklingen som inte ger mervärde till slutkunden. (Hollingsworth, 2011) Ett sämre resultat levererat i tid anses vara viktigare än ett försenat resultat med perfekt kvalitet: ”80 % today is better than 100 % tomorrow” (Ruparelia, 2010).

Viljan att leverera bara de system som ger värde till slutkunden uttrycks i scrum i form av sprints. Som utövare följer man en lista med saker som måste utvecklas utifrån krav från kunden. Resultatet mäts dagligen. (Ruparelia, 2010) Tidsangivelser (tidsestimat) för varje iteration är väsentliga och utgör sprints inom projektet (Mirza, 2010).

Extremprogrammeringen, ofta förkortat XP, är en agil systemutvecklingsmetod som verkligen aktivt gör sig av med designsteget som finns i vattenfallsmodellen. För att leverera värdet för kunden utgår extremprogrammeringen från att kraven ständigt förändras (Moratilla Temprado & Ruz Bendito, 2010) och därför levereras system med små inkrementella förändringar (Ruparelia, 2010).

Vad det gäller kunskapshantering inom systemutvecklingsprojekt koncentrerar sig lean inte på någon praktisk aktivitet för att öka den gemensamma kunskapen utan fokus ligger verkligen på att eliminera onödiga element som inte ger mervärde till slutkunden. Inom extremprogrammeringen arbetar utövaren ofta parvis och arbetsparen byts med jämna mellanrum så att alla ska ha jobbat med alla. Därför sprids kunskapen på ett naturligt sätt inom dessa typer av systemutvecklingsprojekt. Inom scrum existerar daily standups, som är mindre dagsmöten som äger rum på samma plats och på samma tid varje arbetsdag. På dessa möten svarar alla medverkande på frågor om vad som gjorts sedan igår, vad som ska göras idag och vilka hinder som skulle kunna uppstå under dagens arbete. (Moratilla Temprado & Ruz Bendito, 2010) På detta sätt håller sig alla de medverkande i denna typ av systemutvecklingsprojekt uppdaterade om vad andra gör och vad de kommer att göra, vilket ökar kunskapen inom projektgruppen.

Skillnaden mellan lean, scrum och XP låter sig sammanfattas bäst genom att betrakta på vilka abstraktionsnivåer de enskilda systemutvecklingsmetoderna ligger, alltså i vilken grad de är detaljerade och praktiskt orienterade. Lean har en högre abstraktionsnivå och går därmed inte lika mycket in i detalj som till exempel XP. Figuren nedan sammanfattar de agila metoderna enligt deras abstraktionsnivåer:



Figur 2.7: Jämförelse mellan de agila metodernas abstraktionsnivåer. Tagen ur Moratilla Temprado & Ruz Bendito (2010).

2.3.3. Low-technology och gripbarhet

Tillsammans med agila metoder används ofta visualiseringsverktyg såsom kanban och scrumboard, vilka beskrivs i nästa kapitel. Först beskrivs dock low-technology, ibland kallat low-tech, vilket är en beteckning för enkla verktyg såsom pennor, post-it-lappar och whiteboard som används för samarbete och kunskapsdelning. Namnet low-tech anspelar på motsatsen till high-tech, där low-tech betonar just de enkla och "fysiska föremålen". Alajbegovic et al. (2011) ser en stor popularitet av de här enklare verktygen även på högteknologiska (high-tech) organisationer. Skälen till detta är bland annat att de fokuserar på snabbhet och det faktum att problem skapade av high-tech-verktyg ofta kan undvikas.

Ett annat skäl är att det som står på en whiteboard upplevs som "gripbart" (Alajbegovic et al., 2011). Ovan nämndes gripbarhet i samband med redovisningen för skillnader mellan information och kunskap. Kunskap är, som nämnts tidigare, något ogripbart (Nonaka, 1994). Whiteboard, penna och post-it-lappar kan skapa en känsla av kunskapens gripbarhet inom gruppen och detta triggar i sin tur kunskapsspiralen (se figur 2.2).

Ett av de koncept som tar fasta på low-technology-arbetssättet är kanban, vilket beskrivs härnäst.

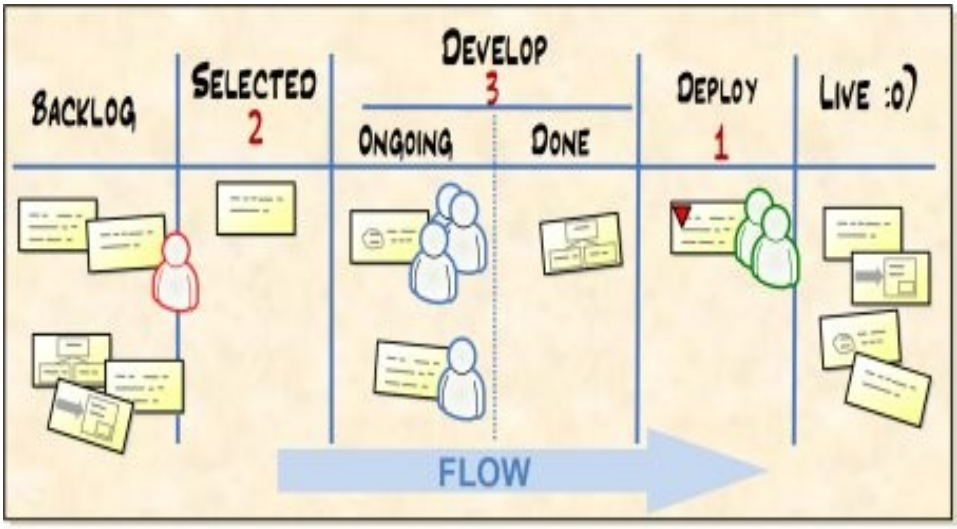
2.3.3.1. Vad är kanban och scrumboard?

En whiteboard är ett verktyg som också används i en annan systemutvecklingsmetod som går under namnet kanban. Kanban, översatt från japanska, betyder bokstavligen ”tavla”. Begreppet härstammar från den japanska bilindustrin, där de anställda istället för att köpa in bildelar med jämna mellanrum gjorde tvärtom, nämligen att de lät arbetarna skriva upp det de behövde på en gemensam tavla.

Denna teknik är också grunden för just-in-time-filosofin inom industrin (Hollingsworth, 2011), det vill säga att producera och leverera produkter i precis den mängd och vid den tidpunkt de behövs (Leander, 2010).

Kanban har sina rötter i lean-metoder, vars syfte är att eliminera onödiga moment i arbetsprocessen som inte ger mervärde till slutkunden (Hollingsworth, 2011). Genom att utveckla bara det som efterfrågas effektiviseras arbetet. Rent organisatoriskt är det bäst genomfört med en tavla, alltså en kanban. En sådan kanban kan vara en fysisk tavla eller en programvara (oftast webbaserad) såsom AgileZen eller LeanKit Kanban. (Hollingsworth, 2011)

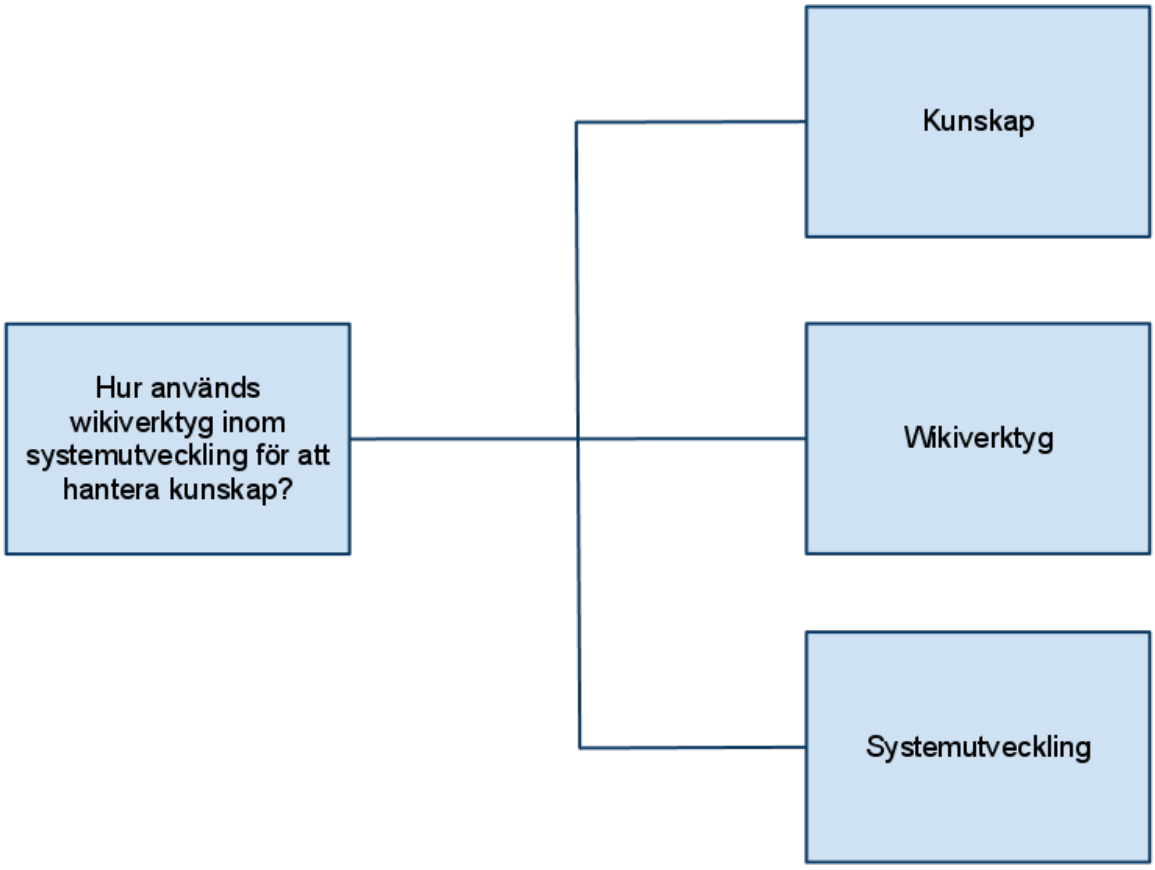
Ett liknande verktyg, som också är närvarande inom systemutveckling, är ett scrumboard. Någon direkt skillnad mellan ett scrumboard och en kanban finns inte utan de är mer eller mindre synonymer till varandra, men de används inom olika systemutvecklingsmetoder.



Figur 2.8: En typisk kanban. Tagen från Crisp.se (2011-03-30).

2.4. Undersökningsmodell

I detta kapitel tas uppsatsens teoretiska ramverk (undersökningsmodellen) fram, vilket är kopplad till teorierna som beskrivits tidigare i uppsatsen. Denna undersökningsmodell har använts för att kunna besvara uppsatsens forskningsfråga som lyder: *Hur används wikiverktyg inom systemutveckling för att hantera kunskap?*. Ur denna forskningsfråga har tre fokusområden tagits fram (kunskap, wikiverktyg och systemutveckling) och dessa har legat till grund för senare kapitel i uppsatsen.



Figur 2.9: Undersökningsmodellen för uppsatsen

I undersökningsmodellen syns forskningsfrågan till vänster medan de tre viktigaste fokusområdena inom uppsatsen är kopplade till denna fråga. När referenser görs i detta kapitel kommer det att vara till tidigare nämna teorier alltså sådana som redan har berörts i litteraturgenomgången.

2.4.1. Kunskap

Detta fokusområde berör både de positiva och negativa aspekterna med hur en wiki används för att hantera kunskap. Tidigare i uppsatsen har det konstaterats att kunskap är ett mångfacetterat ämne som inte har en universell definition (se tabell 2.1) och att det förmodligen finns lika många definitioner på kunskap som det finns människor i världen. I uppsatsen tas det även upp om det är så att det är kunskapen i sig som är svår att dokumentera oberoende av vilket system som används.

Kunskap finns i många former. För att inte blanda ihop kunskap och information är en definition att information är allt utanför vår egen kropp, men så fort en person tar åt sig denna information blir den till kunskap, det vill säga att den konverteras från information till kunskap. (Nonaka, 1994) För att sedan gå djupare ner i definitionerna brukar det skiljas på två typer av kunskap, nämligen explicit och outtalad kunskap. Den explicita kunskapen är sådan som går att skriva ner och formulera så att den blir förståelig för alla. Den outtalade kunskapen däremot är precis tvärtom och den kan vara väldigt svår att formulera med ord. (Nonaka, 1994)

Outtalad kunskap är inte omöjlig att fånga upp, men det kan vara svårt för en anställd på en organisation att förklara sådan kunskap som sitter dennes fingertoppar. Det är därför Nonaka utvecklade en modell för att visa hur kunskap kan konverteras i fyra olika lägen (se figur 2.1). Ett femte läge (Virtual-Socialization) togs dock fram för några år sedan och detta läge förklarar hur kommunikationen och uppfångandet av kunskap kan underlättas med hjälp av diverse IT-verktyg (Spraggon & Bodolica, 2008).

Att fånga upp och hantera kunskap är en komplicerad process. Att då använda sig av en wiki för att genomföra denna process har visat sig vara kostnadseffektivt för organisationen. Det har även visat sig vara enklare att hantera balansen mellan kontroll och flexibilitet inom organisationen när en wiki används. (Jenefeldt, 2010) Det går därför att säga att kunskap skapas, hanteras och sprids oavsett om en wiki finns eller inte, på ett mer eller mindre kostnadseffektivt sätt (Carr, 2007).

2.4.2. Wikiverktyg

Inom detta fokusområde ligger fokus på hur en wiki används idag inom olika organisationer. Huvudsakliga mål har varit organisationer som arbetar med systemutveckling, men även organisationer som inte har systemutveckling som primärt mål har varit av intresse i undersökningen. Utöver wikiverktyg har andra verktyg studerats för att få ett bredare perspektiv i undersökningen.

En wiki har fördelen med att vara webbaserad och att den är öppen för vem som helst att redigera på (Gonzalez-Reinhart, 2005). Organisationer ser dock på möjligheten att kunna begränsa åtkomsten till en wiki, då de inte vill att vem som helst i världen ska kunna ha tillgång till deras interna wiki. Det är bland annat ur detta som konceptet enterprise wiki föddes. En enterprise wiki tar hand om åtkomstproblemen, men behåller även samtidigt en vanlig wikis flexibilitet. En enterprise wiki kan därför sägas hantera balansen mellan flexibilitet och kontroll. (Jenefeldt, 2010)

Ett problem med webb 2.0-sidor, där en wiki faller inom denna kategori (Cormode & Krishnamurthy, 2008), är att den största delen av de som besöker sidan inte bryr sig om att bidra med information utan istället bara tar del av det som andra har bidragit med. Detta kan i sin tur leda till att sidan (eller wikin) blir oanvändbar. (Nielsen, 2006, refererad i Lytras et al., 2009) En undersökning från 2005 visar även att det vanligtvis är några få personer som har bidragit med över hälften av informationen på wikin. (Chau & Maurer, 2005) Detta kan bero på att vissa projekt är för små för att det ska upprättas wikiverktyg inom och därför har de som medverkar inom projektet ingen vana av att dokumentera på detta sätt.

En sak som är viktig att fundera över är hur en wiki ska vara utformad, alltså själva designen. Därför har det tagits fram speciella designprinciper för hur en wiki ska se ut. Dessa användes redan på den första wikin, WikiWikiWeb från 1995. Hur en wiki är uppbyggd påverkar nämligen dess möjligheter att hantera kunskap, så det är väsentligt att fundera över dessa designprinciper när en wiki ska upprättas. (Leuf & Cunningham, 2001, refererad i Müller et al., 2008)

En wiki är ofta lättare att skapa och enklare att redigera än en vanlig HTML-sida. Detta gör att en wiki passar bra som ett sorts socialt nätverk, där användare kan kontakta och hjälpa varandra vid redigeringar av sidan. Det kan alltså skapas sociala band mellan användarna av en wiki. Detta kan även underlättas av att en wiki inte har en hierarkisk struktur utan att den är platt och decentraliserad (Jing & Fan, 2008).

En enterprise wiki kan användas inom många områden inom en organisation, där ett av dessa är systemutveckling. Systemutveckling i sig kan hanteras på många olika sätt och det finns flera olika sorters systemutvecklingsmetoder. I uppsatsen har fokus legat på främst de agila metoderna, som inkluderar bland annat lean (Hollingsworth, 2011), scrum (Ruparelia, 2010) och XP (Moratilla Temprado & Ruz Bendito, 2010).

2.4.3. Systemutveckling

Detta fokusområde behandlar de olika systemutvecklingsmetoderna som används idag av de organisationer som intervjuades i uppsatsen. Fokus ligger främst på de agila systemutvecklingsmetoderna, men även andra systemutvecklingsmetoder tas upp om än som hastigast. Det är detta fokusområde som gör uppsatsen unik, då väldigt få uppsatser tidigare har berört just hur wikiverktyg används inom systemutvecklingsprojekt.

De agila metoderna som behandlas i uppsatsen har alla en gemensam aspekt, nämligen att fokus ligger på snabba leveranser av produkter i mindre iterationer. Systemutvecklingsmetoden lean inriktar sig på att hellre leverera en ofärdig produkt i tid än att leverera en färdig produkt försenat (Ruparelia, 2010). Scrum, en av de andra systemutvecklingsmetoder som tas upp i uppsatsen, syftar till att ha daily standups där projektmedlemmarna uppdaterar varandra om vad de har gjort, vad de kommer att göra och vilka eventuella problem som kan dyka upp under projektets gång (Ruparelia, 2010). Extremprogrammering, eller XP, utgår ifrån att kundens krav ständigt förändras och därmed har denna systemutvecklingsmetod anpassat sig efter detta för att ge kunden mervärde (Moratilla Temprado & Ruz Bendito, 2010).

En av de övriga systemutvecklingsmetoderna som tas upp i denna uppsats är kanban. Det är en systemutvecklingsmetod som tar fasta på low-tech-verktyg, för att informationen (och därmed även kunskapen) ska verka gripbar (Alajbegovic et al., 2011). Inom kanban är det främst en tavla, också den med namnet kanban, som används. Detta är ett low-tech-verktyg (förutom i de fall där kanban är digitaliserat) som används för att lätt visualisera projektets flöde med hjälp av bland annat post-it-lappar. (Hollingsworth, 2011) Det behöver dock inte innebära att kanban utesluter användning av en wiki.

3. Metod och empirisk undersökning

I denna uppsats har en deduktiv metod använts, vilket innebär att vi har utgått från tidigare fenomen och forskning inom området för att sedan skapa egna uppfattningar om ämnet (Jacobsen, 2009). En induktiv metod hade inte varit användbar i denna uppsats, då det redan fanns tillräckligt med forskning inom området för att det inte skulle behövas ytterligare en utforskande uppsats.

Vid den empiriska undersökningen användes semi-strukturerade intervjuer. Vid semi-strukturerade intervjuer finns det fortfarande en lista med frågor som det rekommenderas att man som intervjuare följer såsom vid en strukturerad intervju. Det går dock att ändra på ordningen som frågorna kommer i och som intervjuare får man även lov att ställa oplanerade frågor vid situationer där något kommer upp som är viktigt (Oates, 2006). Vi som författare känner att just intervjuer ger en bra svarsbild på hur läget ser ut idag, då vi anser att intervjuer generellt sett ger en mer nyanserad bild än enkäter. Intervjuguiden som användes i uppsatsen finns som bilaga 2.

I den empiriska undersökningen har vi använt oss av en kvalitativ metod. Även om det finns tre olika sätt att använda sig av den kvalitativa metoden på, intervjuer, observationer och dokumentanalys (Patton, 1987), är det enbart intervjuer vi har valt att fokusera på i denna uppsats.

3.1. Semi-strukturerade intervjuer och urval

Vi som författare gjorde avvägningen att semi-strukturerade intervjuer passade oss bäst vid arbetet med denna uppsats, då vi ibland var tvungna att gå utanför våra förutbestämda frågor och därför att våra frågor hade öppna svarsmöjligheter. Vi höll fem intervjuer, vilka gav oss tillräckligt med information för att kunna dra slutsatser utifrån dessa och då denna mängd intervjuer lämpade sig bäst för oss, då vi använde en kvalitativ metod för vår empiriska undersökning.

Samtliga av de intervjuer vi genomförde skedde ansikte-mot-ansikte, då vi kände att detta alternativ gav mer ärliga svar och därför att det gav oss en chans att studera kroppsspråket på de vi intervjuade. En annan anledning till varför vi valde att genomföra våra intervjuer ansikte-mot-ansikte var att telefonintervjuer, som är ett annat alternativ, ofta är opassande när intervjun består av många öppna frågor (Jacobsen, 2009). Däremot innebär telefonintervjuer lägre resekostnader och minskad risk för en så kallad ”intervjuareeffekt”. Denna effekt innebär att intervjuobjektet uppträder mer onormalt i det fall där en intervjuare är närvarande, alltså att intervjuarens fysiska närvaro påverkar intervjuobjektet (Groves & Kahn, 1979, refererad i Jacobsen, 2009). Dock brukar ansikte-mot-ansikte vara den metod som innebär minst risk mot tillförlitligheten än de övriga alternativen (Jacobsen, 2009) och det är därför vi i slutändan valde detta tillvägagångssätt.

Intervjuerna som genomfördes var alla enskilda, vilket innebär att vi enbart intervjuade en person i taget. Vi som författare anser att enskilda intervjuer är lättare att hantera än gruppintervjuer, även om det finns positiva och negativa aspekter med båda metoderna. Det negativa med gruppintervjuer är att en person kan bli nervös i närvaron av andra personer och därmed inte få fram sin egentliga åsikt (Aubel, 1994). Några andra negativa aspekter är att personer i en grupp har en tendens att hålla med om det som andra säger istället för att uttrycka vad de egentligen tycker och att vissa personer tar en större och mer dominerande roll än andra och därmed inte släpper in de andra personerna i intervjun (Aubel, 1994).

Det finns även positiva sidor till gruppintervjuer och några av dessa är att personerna som blir intervjuade inte känner sig tvingade att svara på alla frågor som ställs. Det brukar även vara så att den som intervjuar får fram mer information på kortare tid än vid en enskild intervju. Slutligen kan gruppintervjuer fostra diskussioner gruppmedlemmarna emellan, vilket i sin tur ger fler synsätt på en specifik fråga. (Aubel, 1994) Vi tog hänsyn till både de positiva och de negativa aspekterna när det gällde gruppintervjuer innan vi gjorde vårt val, där valet i slutet föll på att göra enskilda intervjuer.

Vårt urval av intervjuobjekt rör sig främst om IT-personal som är verksamma i Öresundsregionen, där vi valde att intervjua dessa personer i Lund, Malmö och Landskrona. Detta val baserade vi på att det var möjligt för oss att genomföra ansikte-mot-ansikte-intervjuer, men även då det finns många kunniga personer inom IT i dessa städer. Organisationerna vi har valt ut att intervjua personer på är organisationer som främst sysslar med systemutveckling, men för att inte få ett ensidigt resultat har vi även intervjuat en person på en organisation som inte enbart sysslar med systemutveckling. I alla organisationer vi har varit ute och intervjuat på har vi intervjuat de som har mest kunskap om organisationens IT-uppbyggnad och hur de hanterar kunskap vid deras systemutveckling. Organisationerna i sig är även olika vad det gäller storlek och vi har därför inte begränsat oss till att enbart genomföra vår undersökning på organisationer av en viss storlekstyp.

3.2. Intervjuupplägg

Genom att följa råden i metodlitteraturen (Oates, 2006 och Jacobsen, 2009) har vi som författare utvecklat en agenda för våra intervjuer. Eftersom de flesta av respondenterna hade bokat in en timme för intervju, och vi ansåg att detta var ett bra tidsestimat, planerade vi följande agenda:

Tabell 3.1: Tidsestimat för våra intervjuer:

Cirka tid	Aktivitet	Förklaring
10 min.	Kallprat	Kallprata om väder och om resan till företaget för att ge tillfälle att lära känna varandra och skapa en avslappnad atmosfär. Vi går till rummet där vi sitter och ordnar eventuella praktiska saker.
5 min.	Introduktion	Vi berättar syftet med vår uppsats och förklarar vad vi menar med wiki. Syftet med detta är att ge en bättre bild av vår uppsats och att säkerställa respondentens rättighet att få all nödvändig information som nämns under rubriken Kvalitet (reliabilitet, validitet och etik) (se kapitel 3.4).
10 min.	Uppvärmning	Vi ställer enklare frågor som respondenten med största sannolikheten har svar på. Vi ger ett tillfälle att skapa en bild om oss själva och bestämma hur mycket respondenten kan lita på oss.
30 min.	Huvuddelen	Vi ställer vår uppsats huvudfrågor och genom att vara lyhörda försöker vi fånga upp ytterligare frågor som respondenten kan ha svar på och som kan leda till bättre förståelse inom vårt ämne.
5 min.	Avrundning	Vi ställer enklare frågor igen, bekräftande frågor av typen: “Har vi förstått dig rätt att...?” för att få högre reliabilitet. Inga nya svåra frågor får ställas under denna del av intervjun. Vi tackar och berättar att vi ska skicka en kopia av transkriberingen för bekräftelse.

Intervjufrågorna vi ställde grundade sig i uppsatsens fokusområden från undersökningsmodellen: kunskap, wikiverktyg och systemutveckling (se figur 2.9).

Kunskap

- Hur arbetar ni med kunskapshantering (till exempel skriva ner erfarenheter) inom systemutvecklingsprojekt?
- Beskriv kort om hur ni dokumenterar i era systemutvecklingsprojekt. Är denna dokumentation detaljerad eller generell? Sker denna dokumentation kontinuerligt?
- Vad gör ni för att skapa och bevara kunskap?

Wikiverktyg

Om wikiverktyg används:

- Berätta kort om er wiki.
- Hur länge har wikin funnits?
- Vilken programvara?
- Förklara i vilka sammanhang/speciella projekt wikin används.
- Beskriv några brister med wikin, alltså något som ni tycker kan förbättras med den.

Om wikiverktyg inte används:

- Hur definierar ni en wiki?
- Har wiki(er) använts tidigare? Om ja, vad fick er att gå från wiki(er)? Om nej, har ni planer på att implementera en wiki?
- Vilka samarbetsverktyg använder ni? Täcker detta/dessa program era dokumentationsbehov?

Systemutveckling:

- Hur sköter ni er systemutveckling?
- Förklara vilka systemutvecklingsmetoder ni använder.
- Påverkar valet av systemutvecklingsmetod ert sätt att dokumentera?

Vidare har vi definierat frågor som för samman de tre fokusområdena och dessa frågor kan därför inte beskrivas under enbart en av de ovanstående rubrikerna:

- Beskriv hur ni fångar upp kunskap inom era systemutvecklingsprojekt. Är er wiki till hjälp under denna process (på vilket sätt)?
- Finns det någon kunskap i era systemutvecklingsprojekt som är svår att överföra eller förmedla med hjälp av er wiki?
- Vilka verktyg är då bättre för den typen av kunskap?
- Vilka andra verktyg utöver wikin använder ni för att samarbeta inom systemutvecklingsprojekten? Hur fungerar de ihop?
- Hur är det att utveckla med hjälp av en wiki jämfört med hur ni utvecklade tidigare?
- Hur ser kunskapshanteringen ut i ert systemutvecklingsarbete?

För en komplett intervjuguide se bilaga 2.

3.3. Datainsamling och analysmetod

Vår datainsamling har skett genom semi-strukturerade intervjuer med fem anställda på fyra olika organisationer, där en av intervjuerna var en kompletterande intervju. De vi intervjuade var: Deniz och Tobias på Tretton37, Johan på Syncron, Hampus på Bool och Mikael på Haldex (hädanefter kommer vi enbart att benämna dessa personer vid förnamn förutom i de fall då de uttryckligen nämner sina efternamn). Varför vi valde just dessa organisationer beskriver vi om i kapitel 4.1. Respondenterna valdes av organisationerna själva och ansågs veta mest om det vi hade att fråga dem om.

Intervjuerna vi höll varade mellan 30-60 minuter och enbart i ett av fallen var vi tvungna att komplettera en tidigare intervju. Intervjuerna skedde på respondenternas respektive arbetsplatser.

Intervjuerna spelades alla in, men först efter det uttryckliga godkännandet av respondenterna. Vi som författare valde då även att fråga om den vi intervjuade vill vara anonym eller inte, vilket ingen av respondenterna ansåg vara nödvändigt. Efter själva intervjun påbörjade vi vår transkribering. I transkriberingen valde vi att exkludera allt som inte hade någon relevans till vår uppsats och detta markerades då med ett uteslutningstecken omgivet av två hakparenteser ([...]).

Samtidigt som vi transkriberade påbörjade vi även vår analys av intervjuerna. Analys av kvalitativ data består, enligt Jacobsen 2009, av tre olika faser: beskrivning, systematisering och kategorisering samt kombination. I beskrivningsfasen bearbetas intervjuerna till ett sammandrag (Jacobsen, 2009). Vårt sätt att arbeta här var att transkribera intervjuerna så snabbt som möjligt efter det att intervjun var klar. Transkriberingen ses ibland som det första steget då en analys av de inspelade intervjuerna verkligen sätter fart (Oates, 2006). Under transkriberingen lyfte vi fram vissa av meningarna och kursiverade dessa så att vi lätt kunde återanvända dem när vi skrev resten av vår uppsats. Vi kursiverade även några meningar som respondenterna betonade, då vi ansåg att detta var extra viktig information.

I systematisering- och kategoriseringsfasen är det väsentligt att föra in data i olika grupper vilka ska vara indelade efter specifika ämnen (Jacobsen, 2009). Här skapade vi rubriker som var direkt kopplade vårt ramverk och på detta sätt kunde vi snabbt och enkelt kategorisera respondenternas svar under de passande rubrikerna. Detta låg sedan till grund för senare kapitel.

I kombinationsfasen ska man som författare hitta samband i den data man samlat in (Jacobsen, 2009). Detta gjorde vi genom att skapa tabeller där vi sammanfattade respondenternas svar kopplat till våra fokusområden (se till exempel tabell 4.1).

För att göra vår analys mer förståelig använde vi oss av en kvalitativ analys som beskriver ett stegvis sätt på vilket det går att skapa sig nya uppfattningar under analysens gång och där analysen visar sig vara en iterativ process (Dey, 1993). Vi kände att det var så vi arbetade med vår analys, då vi redan under intervjuerna skapade oss en uppfattning om resultaten, men att vi senare under transkriberingen fick nya insikter om andra tolkningssätt och att vi på det sättet använde oss av metodens iterativa process för att slutligen nå vårt resultat.

3.4. Kvalitet (reliabilitet, validitet och etik)

Detta kapitel beskriver hur pass pålitlig den data är som vi som författare har samlat in (reliabilitet), om vi undersökte det vi ville undersöka (validitet) och om vi följde de (oskrivna) regler som finns i samhället idag (etik).

För att minska reliabilitetsfällorna har vi haft det i medvetandet att förhålla oss objektiva så gott det går under intervjuerna och att inte ställa vinklade frågor. Efter transkriberingen har vi bett respektive respondent att läsa intervjun och se om allting uppfattades på ett korrekt sätt. Själva transkriberingsarbetet var också ett steg att tänka efter och skapa en mer objektiv bild (Oates, 2006).

Vi har använt oss av vetenskaplig litteratur med en review-/granskningsprocess för att kunna lita på de teorier vi har beskrivit i uppsatsen. Dessa har sedan använts för att utforma intervjuguider och analysera insamlad data. Vi har även förhållit oss källkritiska i den mån att vi alltid försökt ta oss till ”basreferensen” för att förstå vad den ursprungliga författaren egentligen ville ha sagt med det han/hon hade skrivit. De etiska aspekterna som är kutym i samhället idag följde vi när vi skrev denna uppsats, det vill säga att vi inte kränkte eller tvingade några individer att ställa upp på något de inte frivilligt ville.

Oates (2006) utgår från dessa rättigheter som ska säkerställas för undersökningens (i vårt fall intervjuernas) deltagare:

1. Rätt till att låta bli att delta.
2. Rätt att avbryta deltagande.
3. Rätt att få all information om undersökningen såsom syfte, skribenter, hur data ska användas, på vilket sätt de blir involverade med mera.
4. Rätt att vara anonym, det vill säga att kräva att deras namn blir oigenkännliga.
5. Rätt att kräva tystnadsplikt och sekretess, vilket innebär att de ska kunna begära av skribenten att all data behandlas som hemlighetsstämplad och det garanteras att denna data inte läcks ut.

I vår uppsats har vi utgått från de etiska principerna som beskrivits ovan för att inte kränka någons integritet. Vid varje intervju har vi gett all information om vår undersökning som behövts och som respondenterna velat ha. Vi har även påpekat att deras deltagande är helt frivilligt och kan avbrytas när som helst. Vi har även frågat intervjupersoner om de önskar vara anonyma, bli behandlade med tystnadsplikt och/eller sekretess.

För att försäkra oss om att vi inte uppfattat respondenternas svar fel och på det sättet ofrivilligt förvrängt eller manipulerat insamlad data har vi, efter att ha gjort transkriberingen, gett ett exemplar till respektive respondent för godkännande och kontroll.

Vi genomförde totalt fem intervjuer, vilket inte behöver ge ett tillräckligt vetenskapligt bevisat resultat och på grund av detta undviker vi att dra generella slutsatser i onödan.

4. Empiri och resultat

I detta kapitel beskrivs först och främst de organisationer som intervjuerna har genomförts på. I delen som benämns empiriska resultat sker en djupdykning i vad respondenterna på dessa organisationer har sagt och en sammanställning i tabellform avslutar kapitlet.

4.1. Undersökningsobjekt

Personliga semi-strukturerade intervjuer har genomförts med fem personer på fyra organisationer: Tretton37 i Lund, Syncron i Malmö, Bool i Lund och Haldex i Landskrona. På Tretton37, som inte har någon in-house-systemutveckling för närvarande, genomfördes två intervjuer: en ordinarie med VD:n Deniz och en kompletterande intervju med konsulten Tobias, som berättade om wiki-användning vid systemutveckling ute på uppdrag.

Varför vi som författare valde just dessa organisationer var på grund av några olika anledningar. För det första valdes Tretton37 och Bool därför att de är snarlika organisationer, det vill säga de är relativt små och båda har systemutveckling som huvudfokus. Dock skiljer de sig åt genom att Tretton37 enbart sysslar med .NET och Bool enbart med Sharepoint. På så sätt har vi fått varierande svar från de båda organisationerna. Syncron valde vi därför att de var en något större organisation, men de har ändå systemutveckling som en av de viktigaste hörnstenarna. De har även varit verksamma på marknaden längre än vad både Tretton37 och Bool har varit, vilket ger oss en inblick i en IT-organisation som har överlevt några tuffa perioder och som även har genomgått flera olika organisatoriska förändringar. Slutligen valde vi Haldex, då vi ville få inblick i en organisation som inte har systemutveckling som kärnaktivitet, men som ändå till viss del arbetar med detta. Haldex är även både den äldsta och största av organisationerna vi intervjuade och därmed fick vi en inblick i hur en så pass stor organisation hanterar dokumentationen av kunskap.

För kortfattade presentationer av de fyra organisationerna, se bilaga 5.

4.2. Empiriska resultat

Forskningsfrågan *Hur används wikiverktyg inom systemutveckling för att hantera kunskap* är indelad i tre fokusområden: kunskap, wikiverktyg och systemutveckling. Dessa fokusområden ligger till grund för rubrikerna i detta och nästkommande kapitel.

4.2.1. Kunskap

Som nämnts tidigare ligger fokus inom detta fokusområde på de som använder sig av wikin, det vill säga i vårt fall systemutvecklarna. I undersökningen sågs det på vilka behov och intressen som dessa personer hade, men även deras egna åsikter om wikin. En annan vinkling som var av stort intresse för undersökningen var hur wikin kunde stödja samarbetet inom systemutvecklingsprojekten.

Tobias kände sig osäker på om det var wikin i sig som främjade uppfångandet och nedskrivandet av kunskap eller om detta gjordes på grund av att projektmedlemmarna hade ett intresse för det. Han tyckte dock att det var behovet av att dokumentera inom projektgruppen som låg till grund snarare än att det var själva wikin som fick folk att redigera i den. Han nämnde även att det som skrevs ner i wikin var saker och ting som ”gick på rutin”, det vill säga kunskap som är viktig för alla inom projektet.

Johan konstaterade att det finns kunskap som är svår att föra in på wikin. Just mjuka kunskaper, såsom känslor och åsikter, tyckte Johan var svåra att få dokumenterade på wikin, men att detta inte berodde på wikin i sig utan att det skulle vara lika svårt att skriva ner i till exempel ett Word-dokument. Han sa även att om det gick att formulera i text så gick det att skriva ner på wikin. En annan sak som var svår att få in på wikin, enligt Johan, var bilder. Det fanns inte någon direkt möjlighet att rita direkt i wikin utan att använda vissa tilläggsprogram, så detta var ännu en sak som inte wikin var optimal för, enligt Johan.

Hampus berättade att de brukar börja med den generella informationen och sedan specificera den allteftersom. Det är mestadels systemdokumentation som skrivs in, men det finns ingen klassificering av det som läggs ut. Dessutom läggs dokumentationen ut kontinuerligt. Hampus tyckte att problemet låg i att skriva ner allt det som inte var tekniskt. Alla de mjuka delarna (i detta fall kundkontakt och kunddialog) tyckte han var väldigt svårt att få ner i ord.

Mikael förklarade att deras sätt att dokumentera grundade sig i vilken typ av systemutveckling de sysslade med. De skriver därför inte ner någon grunddokumentation över de olika systemen utan de använder den som finns tillgänglig från leverantören. Därför blir det mesta av dokumentationen som skrivs ner generell. Någon specifik systemdokumentation existerar nästan inte mer inom Haldex. Kunskapen delas istället med hjälp av .pst-filer (mailarkiv-filer i Outlook) och det tvekas inför varje gång det måste bytas ägare på dessa filer då det är svårt att hitta i någon annans struktur, förklarar Mikael.

4.2.2. Wikiverktyg

Vad det gäller detta fokusområde var det främst organisationen som var i fokus. Vi som författare försökte se på vilka förutsättningar organisationen har att bibehålla kunskapen inom organisationen, men även hur kunskapen skapades och underhölls. Det undersöktes även om det var så att det var kunskapen i sig som var svår att dokumentera oberoende av vilket system som användes.

När vi bad respondenterna att beskriva några brister med deras wiki och även om detta fick användarna att känna obehag inför wikin berättade Tobias att han inte kunde se några direkta nackdelar med att använda en wiki för att dokumentera utan att problemet istället låg i att få folk att dokumentera i allmänhet. Han ansåg att problemet med att få folk att dokumentera låg i hur folk var "funtade", inte hur wikin var uppbyggd.

Johan berättade om några mindre nackdelar med wikin, såsom att det inte gick att söka med en inledande "stjärna" (*) och att själva redigeringen av en wiki-sida inte alltid blev som man som användare ville. Han ansåg dock inte att detta fick folk att känna obehag inför wikin utan snarare att det bara retade folk lite grann. Han poängterade även att så länge det fanns de två olika redigeringssätten (som han benämnde "wiki-markup-läget" och "Word-läget") på wikin trodde han att folk skulle använda den.

Hampus ansåg att problemet med wikin låg på att det tog för mycket kraft att dokumentera på den. Att dokumentera på wikin tog, enligt Hampus, "fem minuter av ens extremt värdefulla tid", men samtidigt är han osäker på om det finns något bättre sätt att lösa dokumentationsproblemet på än med hjälp av en wiki.

På frågan hur det är det att utveckla med hjälp av en wiki jämfört med hur det utvecklades tidigare tyckte Tobias att wikin hade några stora fördelar såsom att det är ett lätt medium att arbeta och redigera texter i. Han tyckte även att det var en större frihet att arbeta med en wiki, då det inte finns några begränsningar i hur du utformar dokumentet från början. Tobias tyckte även att det var en frihet att få sätta de standarder man själv ville istället för att använda fördefinierade sådana.

Johan ansåg att det är smidigare att dokumentera nu, med hjälp av en wiki, än det var tidigare. Han berättade att de hade fått arbeta med ärendehanteringssystem ute hos sina kunder och dessa system var inte lika spridda som wikin var. Sedan var det känsligt att skriva där också, då man som konsult inte ville skriva exakt hur man gjorde i kundens egna system utan man ville hellre skriva internt och detta underlättades när Syncron började använda sig av Confluence, deras interna wiki, förklarade Johan.

Hampus förklarade att de använde sig av ett pärmsystem i början, men att de har gått över till projektsajter och användandet av en wiki nu på senare tid. Just wikin tycker han inte ersätter de gamla pärmsystemen, men att wikin tillsammans med en specifik projektsajt fyller de dokumentationsbehov de har. Han förklarade även att det är nu under det senaste året som wikin har kommit till därför att organisationen expanderat starkt under denna tid. Han uppskattade att wikin gjorde det möjligt för fler personer att ta del av den kunskap som skrevs ner på den, vilket inte var läget när de bara använde pärmsystemet.

Mikael berättade att de aldrig har använt sig av en wiki på Haldex, men att han personligen har arbetat med wiki på annat håll tidigare och då hade han enbart goda erfarenheter av det. Han förklarade att en wiki enbart kan överleva om användarna föder den med information och om den engagerar folk. Vidare sa han att det inte vore en dum idé att implementera en wiki på Haldex, då den skulle kunna samla och strukturera informationen på en plats istället för att informationen ska finnas på många olika platser som det ser ut idag.

4.2.3. Systemutveckling

Inom detta fokusområde var det främst systemutvecklarna som var i fokus. Vi som författare försökte se på om de olika systemutvecklingsmetoderna hade någon inverkan på sättet som systemutvecklarna dokumenterade på och vad det fanns för substitut till wiki inom de olika organisationerna.

På frågan om valet av systemutvecklingsmetod påverkade sättet att dokumentera svarade Tobias på Tretton37 att han inte trodde att det var så. Han pekade istället på att det låg hos beställaren att ställa krav på vad som skulle dokumenteras. Han ansåg även att det som dokumenterades skulle ha ett syfte. "Man ska inte dokumentera för dokumenterandets skull" var hans åsikt.

Vid intervjun med Johan på Syncron hade han inget rakt svar på om valet av systemutvecklingsmetod påverkade hur dokumentationen gick till. Han ansåg att systemutvecklingsmetoden scrum brukade beskyllas för att inte bry sig om dokumentationen, men att han inte tyckte att det stämde riktigt. Han tyckte att scrum istället har minskat på mängden dokumentation för att systemutvecklingen skulle ledas av systemutvecklare. Johan har jobbat i projekt som har använt scrum och projekt som har använt vattenfallsmodellen och i alla projekt har det varit underförstått att det är viktigt att ha en bra dokumentation.

Hampus på Bool tyckte att det skulle skiljas på om det gällde dokumentation till kund eller enbart intern dokumentation. Om det gällde att dokumentera till kund ansåg han att det inte spelade någon roll vilken systemutvecklingsmetod som användes. Däremot när det gällde att dokumentera för eget bruk tyckte han att systemutvecklingsmetod och viljan att dokumentera hängde ihop. Han berättade att när agilt arbete utförs är det främst i slutet av delprojekten som det finns tid att reflektera över vad som gjorts under de senaste veckorna och att det först är då som det blir dokumenterat.

Vid vår intervju med Mikael på Haldex ansåg han att det fanns en koppling mellan systemutvecklingsmetod och viljan att dokumentera. Han särskilde på två typer av systemutveckling: nyutveckling och förändringsutveckling. På Haldex arbetar de främst med det senare och Mikael konstaterade att om de hade arbetat med nyutveckling hade behovet av bra dokumentation varit större än det var i dagsläget. Han påpekade även att systemutvecklingen måste anpassas till den verklighet man befinner sig i och att det generellt sett har funnit en ovilja mot mer administration och kostsam dokumentation.

När respondenten skulle förklara i vilka sammanhang/speciella projekt som wikin användes valde Tobias att förklara att wikin som han använde inte hade med något speciellt projekt att göra utan att den snarare var till för de som arbetade med systemutvecklingen. Det som skrevs ner i wikin handlade om saker såsom att beskriva komponenterna som används, informationen för hur dessa fungerar och installationsprocesserna och inte dokumentation runt själva projektet. Han sa att beslutet runt detta låg i att det var intressenternas vilja att det skulle vara uppdelat på detta sätt, men att han gärna hade skrivit upp projektdokumentationen i wikin om han själv hade fått välja.

Johan berättade att wikin används inom nästintill alla projekt förutom vissa kundprojekt, även om kunderna ibland har tagit åt sig fördelarna med att dokumentera på en wiki. Johan har arbetat med att få alla involverade i projektet att dokumentera allt berörande dessa projekt i deras wiki som heter Confluence. Han sa att de främsta fördelarna med detta låg i att det är lätt att spåra informationen och att användarna slipper gissa var den finns någonstans.

Hampus konstaterade att deras wiki användes i alla projekt, då den inte var knuten till något speciellt projekt. Ingen projektspecifik dokumentation skrevs in på wikin utan det dokumenterades väldigt generellt där, det vill säga hur olika problem löses och vad som ska tänkas på när dessa problem uppstår.

När frågan ställdes om vilka andra verktyg utöver wikin som används för samarbete inom systemutvecklingsprojekten svarade Tobias att intressenter inom projekten avgör vilka verktyg som lämpar sig bäst att använda. En viktig intressent är beställaren som kan ställa krav på bland annat dokumentationen. Den organisation Tobias arbetade på för tillfället använde "hederliga Word-dokument" för dokumentation, för att det passar in i organisationens övriga arbete med dokumentation. Medan wikin passar bäst som ett internt verktyg för att skriva upp installationsprocesser och komponentbeskrivningar inom systemutvecklargruppen skrivs systemdokumentationen i Word som lämpar sig bäst för beställaren, men är mindre smidigt för systemutvecklarna eftersom det innebär merarbete att manuellt fylla i information såsom revisionsnummer, vem som har skapat dokumentet, vem som uppdaterat det, när detta skedde och liknande. En wiki lämpar sig mindre bra i Tobias grupp även för deras arbetsflöde. Uppgifter från backloggen skrivs upp som post-it-lappar i ett digitalt scrumboard, som är baserat på Team Foundation Server. Medan uppgifter i backloggen kan ligga till grund för framtida dokumentation ska alla post-it-lappar, vare sig fysiska eller digitala, försvinna när en uppgift är färdig, då dokumentation om uppgiften ska flyttas till koden.

Johan påpekade vid flera tillfällen att han vill få in så mycket som möjligt på deras enterprise wiki Confluence. På det sättet kan alla få tillgång till informationen oavsett vart dessa befinner sig i världen eller hur länge de har jobbat på organisationen. Tillsammans med wikin används andra system, som lämpar sig bättre för vissa ändamål än vad wikin gör, såsom ärendehanteringssystemet Jira och versionshanteringssystemen för kod, som går under namnen CVS (Concurrent Versions System) och Subversion samt representationsverktyget för versionshantering, som kallas Fisheye. Förutom detta använder sig organisationen av ett verktyg för automatisk kodkompilering, vid namn Hudson. För dessa funktioner saknas motsvarighet i wikin. Däremot är de alla integrerade i wikin och det går att se innehållet ur alla dessa system i de andra verktygen. Som nämnts tidigare använder sig Syncron av tilläggsprogram för att hantera bilder i wikin. Detta program går under namnet Gliffy och fungerar väldigt bra tillsammans med Confluence, enligt Johan.

Även Syncron använder scrum som systemutvecklingsmetod, vilket innebär att det används en tavla för arbetsflödet, en så kallad scrumboard, där det finns en fysisk scrumboard i kontoret i Warszawa. Även om scrumboard lämpar sig bra för att visualisera arbetsflödet, enligt Johan, används den dock inte primärt eftersom en Business Analyst kan sitta i Malmö och projektintressenter kan befinna sig i till exempel Tokyo, London eller någon annanstans där kunden är. Johan fortsatte med att berätta att principen vad det gäller scrumboards och wikiverktyg inom organisationen är att "man ska inte göra någonting som inte finns dokumenterat på Confluence". Så det är därför det börjar komma extratillägg för wikin som kan hantera scrumboards. Inom organisationen finns det även direktkommunikation som ingen dokumentation kan ersätta. Syncron använder därför Skype och delvis telefon för all direktkommunikation. Uppföljningar av ärenden kan även göras med hjälp av mail. En annan situation som Johan tar upp, som förvisso finns i deras enterprise wiki fast i ett embryo-tillstånd, är nyhetsflöden och dess bevakning.

Allt som görs i wikin skapar nyhetsflöde som senare visas upp där och detta tycker Johan är till stor hjälp. Han går vidare med att säga att det som är mindre bra är att det inte sorteras. Alla får all information, vilket gör att det lätt kan tappas viktiga nyheter. Johan nämner därför ett verktyg som heter Asana, där det går att välja sina projekt, lägga till dessa som sina favoriter och på det sättet få bara de nyheter som man som användare är intresserad av. Dock använder sig inte Syncron av Asana i dagsläget.

Hampus berättar att inom deras organisation används wikin främst i projektövergripande sammanhang för att systemutvecklarna ska kunna dela med sig av kunskaper om systemutveckling. Bool har sin egen systemutvecklingsmetod som heter Boolean. I denna metod används en tavla för arbetsflöde, en så kallad kanban. Boolean med kanban är något som varken wiki eller något annat verktyg kan ersätta och är enligt Hampus "nyckelframgångsfaktorerna för Bool". På denna kanban finns det fysiska post-it-lappar som har olika färger för olika prioriteringar och detta är enligt Hampus ett "extremt tydligt [och] konkret" sätt att arbeta på.

Alla uppgifter från backloggen bryts ner, visualiseras och tilldelas någon systemutvecklare beroende på hur det ser ut med resurser. Varje projekt på Bool får en egen sajt i organisationens intranät som bygger på Sharepoint. I denna sajt sparas alla kravspecifikationer, all kommunikation med kunden och alla andra dokument som produceras under projektets gång.

Eftersom det produceras information främst i dokumentformat under projekten passar en Sharepoint-sajt bäst, då med möjligheten att ladda upp filer till ett dokumentbibliotek, säger Hampus. Även på Bool spelar direktkommunikationen en stor roll. Systemutvecklarna som sitter på kontoret kan alltid fråga varandra direkt, men förutom detta används Office Communicator, som är en chatt där systemutvecklarna kan skriva till varandra i realtid.

Mikael berättar att de använder Outlook, Word och Office Communicator en hel del inom Haldex. Det används även en Sharepoint-baserad portal internt på ICT-avdelningen. Eftersom Haldex inte har någon wiki kunde inte Mikael besvara frågor om situationer där andra verktyg lämpar sig bättre.

Däremot erkände Mikael att de tittar på olika alternativ för att förbättra dokumentationen inom organisationen, speciellt med tanken på att Haldex mer och mer har blivit en multinationell organisation, vars tillverkning finns i bland annat Ungern och Mexiko. Eftersom det inte sker någon nyutveckling inom organisationen handlar det mycket om anpassning och förbättring av befintliga system på Haldex. Därför är ärendehanteringssystemet (här kallat "casesystem") HEAT centralt. Problemet med det systemet är det inte går att söka om som användare inte känner till ärendenumret. Vad det gäller nya förfrågningar i systemet (så kallade "Request for Change") dokumenteras dessa i Word-dokument, som fylls i av beställaren och av utföraren. Dessa dokument skrivs ut och sparas i pärmar, men de ligger även kvar i de involverades Outlook-mappar.

När följdfrågan hur bra integrerade alla dessa system är med wikin ställdes svarade Tobias att de använder en Sharepoint-wiki, vilken i sin tur är en del av Microsoft-miljön och systemen går därmed bra att integrera. I övrigt är det inga andra kopplingar mellan wikin och andra verktyg som Tobias kunde komma på.

Johan förklarade att deras ärendehanteringssystem Jira och deras representationsverktyg för versionshantering, Fisheye, fungerar bra ihop med Confluence. Alla de här verktygen är nämligen skapade av samma organisation, som heter Atlassian, och integreringen är därmed redan inbyggd. I och med att det på en wiki går att ladda upp vanliga dokument och bilder finns det möjlighet att komplettera texten med skärmdumpar och Visio-ritningar. Det som Johan ser med glädje på är att det utvecklas nya tillägg för Confluence såsom möjligheten att rita direkt i wikin. Även själva Confluence uppdateras ständigt och jämt och ger fler möjligheter att samarbeta.

Hampus beskriver deras wiki i ett portalsammanhang. Bools wiki bygger på Sharepoint och är integrerad med intranätet samt hela organisationens infrastruktur, som är standardiserat på Microsoft-plattformen. Hampus berättar att alla nya uppdateringar på wikin kommer direkt som nyheter i portalen och vid varje namn (alltså den som har skrivit) finns det en ikon som signalerar om ens status (om man är online, upptagen eller offline). Genom att klicka på namnet kan man skriva ett chattmeddelande, skicka ett mail eller starta ett telefonsamtal.

Mikael berättar att det hade varit bra att ha ett system för dokumentation och kommunikation och "varför inte en wiki?". Detta är något som kommer göras senare i så fall. För tillfället kunde han inte säga något om ifall det skulle gå att integrera wikin med andra system utan några större problem.

Vid frågan om dessa andra verktyg täcker organisationens dokumentationsbehov svarade Tobias att wikin bara används för en specifik sak, nämligen att skriva ner projektövergripande information inom systemutvecklingsteamet. Det är snarare så att wikin kompenserar det som andra system inte kan ge eftersom wikin är ett smidigt sätt att skriva ner och spåra ändringar.

Johan talar om avsaknaden av en vettig funktionalitet för att rita direkt i wikin. Detta kompenseras med Visio och skärmdumpar. Extratillägg i wikin för att rita, såsom Gliffy, håller på att utvärderas.

Hampus ger också wikin en mindre roll inom organisationen. Liksom i Tobias fall, på Tretton37, används wikin snarare som ett kompletterande verktyg.

Mikael, som inte använder någon wiki, berättar om olika sätt för kunskapshantering. Avsaknaden av ett dokumentationssystem kompenseras med en fileshare där olika Word-dokument ligger och där dessa dokument även finns i pärmar. Det skrivs många mail och all mailkommunikation sparas och sorteras i olika mappar. Mikael säger att det de behöver är en tydligare spårbarhet, som bara delvis går att åstadkomma med mail-mappar och .pst-arkivfiler, vilka lämnas över till efterträdaren vid ett eventuellt jobbyte.

Ytterligare en sak som en wiki gör är att den i princip skapar en plats där all nödvändig information finns och där denna information är sökbar och tillgänglig för alla behöriga. Mikael påpekar att det finns massor med ställen där informationen kan finnas. För att ta reda på en specifik sak kan han tvingas kolla i pärmar, i sin mail, i filesharen, i casesystemet HEAT, i föregångares .pst-arkivfiler och även i källkoden. Om han ändå inte hittar den information han söker ringer han en kollega eller till organisationen som utvecklar systemen åt Haldex.

Tabell 4.1: Intervjuresultat: Hur används wikiverktyg inom systemutveckling för att hantera kunskap?

Fokusområden	Tretton37	Syncron	Bool	Haldex
Kunskap	Kunskap och kunskapsdelning är centrala för organisationen. Ständig utveckling är nyckeln till framgång. Olika forum internt och externt i communityn arrangeras.	Kunskap är centralt för denna IT-organisation. Både kunskap inom teknik, programmering och industri, men även praktisk kunskap i olika projekt och organisation.	Bools specialisering ligger kring samarbets-plattformen Sharepoint. Djup kunskap om utveckling för Sharepoint och kunskap om kundernas behov är centralt för Bool.	Kunskap finns utspridd i olika pärmar, fileshares, Outlook-mappar, i kodkommentarer och i ärendehanterings-systemet HEAT. Mycket av informationen ligger offline och kunskapen delas vid förfrågan.
Wikiverktyg	Wikiverktyg används inte på organisationen, men förekommer ofta på konsultuppdrag. Wikiverktyg är ett lätt sätt att skriva ner och spåra ändringar. Först och främst använder de wikiverktyg för att skriva upp ”saker som går på rutin”.	Organisationens wiki, Confluence, är en central plats för all informations- och kunskapsdelning. Wikin används som informations-överföring, men även som kommunikations-forum för diskussioner mellan anställda i olika länder.	Organisationen använder en ”teknikwiki” där det skrivs upp problem och lösningar. Det ska gå snabbt och enkelt. Teknikwikin är en ”brain dump” och är inte kopplad till något specifikt projekt.	Det finns embryon till wikiverktyg i form av en ICT-sajt, byggd på Sharepoint. Den används endast inom IT-avdelningen för tillfället. Mikael tror att det går att få gehör bland övriga inom organisationen för något sorts samarbetsverktyg.

Fokusområden	Tretton37	Syncron	Bool	Haldex
Systemutveckling	Agila metoder föredras såsom metoden scrum. Det är kunden som bestämmer i slutändan. Det finns ingen speciell systemutvecklings-metod som dominerar. Tretton37 använder ingen egenutvecklad metod heller, då det kan vara svårt att ta med den till kunden. Även att använda någon metod till punkt och pricka är omöjligt. ”ScrumBut” används enligt Tobias.	Numera används scrum fast det kan variera från projekt till projekt. System-utvecklingen är kopplad till organisationens wiki: Confluence. Principen är att ”ingenting görs förrän det finns i wikin”, som Johan sa. Versions-hanterings- och ärendehanterings-system används flitigt, vilka då är integrerade med wikin.	Organisationen är ung och har ett agilt tänk. Scrum har körts en del, men har visat sig inte passa helt in i organisationen. En egenutvecklad metod, Boolean, har tagits fram som bygger på agila metoder och lean software development. Fokus ligger bland annat på visualisering och tydlighet.	Framförallt förändrings-utveckling sker på Haldex. Det finns en stor uppsättning av gamla applikationer som måste skrivas om. De och gamla systemet Axapta tillåter inte användning av nya metoder och moderna verktyg. Alla gamla versioner av kod markeras ut och nya kommenteras direkt in i koden. Det finns en stor önskan att förnya systemen.

5. Analys och diskussion

I detta kapitel analyseras de resultat som togs fram i föregående kapitel. Innan analysen och diskussionen påbörjas vill vi som författare förklara några märkbara skillnader organisationerna emellan:

Tretton37 är helt och hållet en konsultorganisation som har sina medarbetare utspridda hos sina kunder, men som ändå värdesätter kommunikation och kunskapsspridning inom sin splittrade grupp.

Syncron är en global, men relativt liten, organisation med cirka 120 anställda som utvecklar sin egen programvara för bland annat produkthantering. Syncron gör allt för att upprätthålla tillgängligheten för alla involverade oavsett tidszon, språk och kultur.

Bool utvecklar anpassade lösningar av Sharepoint-baserade portaler, mestadels in-house, genom att använda sin egen metod Boolean som stammar av scrum, lean och kanban.

Haldex skiljer sig från de tre övriga organisationerna som intervjuats i och med att utveckling av programvara inte är deras huvudsysselsättning. Det är dock ett medvetet val från vår sida att samla empiriskt material från olika typer av organisationer. Haldex är en industriorganisation som inte direkt sysslar med IT, men är beroende av ett fungerande IT/IS-stöd. Utveckling och anpassning av affärssystemet Axapta sker kontinuerligt och det har varit mycket intressant att se hur dokumentationen och kunskapshanteringen ser ut på ett organisation utan wiki, där det inte är möjligt med direktkommunikation såsom det är möjligt på de mindre organisationerna.

5.1. Kunskap

Tobias på Tretton37 kunde inte svara på om det var wikin i sig som gjorde att projektmedlemmarna dokumenterade på den eller om det var på grund av deras genuina intresse för dokumentation. Han konstaterade istället att det var behovet av att dokumentera som gjorde att projektmedlemmarna skrev in saker i wikin och att det som skrevs in var de ”saker som gick på rutin”, det vill säga den outtalade kunskapen inom projektet. För att få del av denna kunskap kan det därför krävas av en ny projektmedlem att konvertera denna outtalade kunskap till sin egen, antingen genom Socialization- eller Externalization-lägena (Nonaka, 1994) eller det som Newman och Conrad kallar för knowledge creation och knowledge transfer (Newman & Conrad, 2000).

Mikael däremot, som arbetade på en organisation som inte använde en wiki i dagsläget, förklarade att deras sätt att dokumentera hade sin grund i vilken systemutveckling de sysslade med och nämnde alltså inte om det fanns någon sorts kunskap som var svårare att dokumentera än annan.

Det han sa var dock att kunskapen delades via .pst-filer (mailarkiv-filer i Outlook) och att det tvekas inför varje gång dessa tvingas byta ägare, då varje person har sin egen struktur i Outlook.

Detta är ett exempel på Internalization, där en person konverterar någon annans explicita kunskap, i detta fall struktur i ett program, till sin egna kunskap, det vill säga att den nuvarande strukturen görs om till något som passar en själv (Nonaka, 1994). Denna egna, outtalade kunskap blir i sin tur explicit den dagen då man själv tvingas ge från sig sina .pst-filer och den har då gått igenom ännu ett läge av kunskapskonverteringen, nämligen Externalization (Nonaka, 1994) eller som Newman och Conrad kallar det: knowledge transfer (Newman & Conrad, 2000).

Både Johan och Hampus var överens om att det fanns kunskap som var svår att dokumentera. Johan nämnde mjuka värden, alltså känslor och åsikter, som exempel och att svårigheten att dokumentera dessa inte var på grund av wikin utan att dessa hade varit lika svåra att formulera i ett Word-dokument. Han nämnde då att om det går att få ner i text går det att få in på wikin. Hampus hade en liknande åsikt som Johan, fast han nämnde kundkontakter och kunddialoger istället för känslor och åsikter. Hampus sa även att allt som inte var tekniskt var svårt att dokumentera på wikin. Båda dessa påpekanden är klassiska exempel på att outtalad kunskap är svår att konvertera till explicit, vilket Externalization-läget (Nonaka, 1994) och knowledge transfer-aktiviteten (Newman & Conrad, 2000) går ut på.

Vår slutsats blir därför att mjuka värden, såsom känslor, åsikter och kunddialoger, är svåra att dokumentera oavsett om en wiki finns till hjälp eller inte.

Vad det gäller hur wikin skiljer sig från det som användes innan för att sköta dokumentationen, var alla vi intervjuade överens om att en wiki gör det lättare för användarna att sprida information. En sak som togs upp som fördel var att wikin var lättillgänglig och att det på ett enkelt sätt går att samla all nödvändig information i den. Dock poängterade Mikael att en wiki inte underhåller sig själv utan den måste engagera folk, för att få dem att fylla på wikin med information.

De som använde sig av en wiki på sina respektive organisationer höll med om att wikin har underlättat för dem i deras dokumentationsarbete. Tobias ansåg att en av de största fördelarna med en wiki var att det inte fanns några begränsningar i den. Johan uppskattade det faktum att han slapp skriva in all dokumentation i kundernas interna system när han använde sig av en wiki. Tidigare hade Johan använt sig av programmet Inside, vilket var kopplat till Sharepoint, men han tyckte att deras nuvarande lösning med Confluence som wiki var en mycket bättre sådan. Hampus tyckte att det var förnämligt att wikin var så central som den var, men han tyckte ändå inte att den kunde ersätta de fysiska pärmar som organisationen använde sig av.

Mikael, som arbetar på en organisation som inte använder sig av en wiki, tyckte att en wiki lät som en bra lösning för att kunna centralisera och strukturera den information som organisationen har på flera olika ställen i dagsläget.

De aspekter som tagits upp i dessa stycken hör samman med de designprinciper som vi har beskrivit tidigare, som visar på att en wiki är öppen, simpel, organisk, universell och delbar (Müller et al., 2008).

Vår slutsats blir därför att en wiki underlättar vid dokumenterandet inom systemutvecklingsprojekt, men att den inte nödvändigtvis behöver ersätta äldre lösningar.

Det finns verktyg som är specialanpassade för vissa uppgifter inom systemutvecklingen såsom versionshantering och ärendehantering. En wiki kan inte lösa alla problem som uppstår under utveckling av system. Om det finns en bra integration mellan wiki och andra verktyg känns dessa verktyg som en del av wikin, som vi ser i Johans fall på Syncron.

På Bool och Tretton37 används wikin för en relativt liten del av systemutvecklingen, där de skriver ner projektövergripande information om till exempel installationer och infrastruktur. Bool använder wikin som en brain dump för ny kunskap, som systemutvecklare har skapat, det vill säga att det skrivs ner vilka problem som uppstod, möjliga lösningar på dessa, vilken ny teknik som finns tillgänglig och så vidare. Wikin i sig på Bool och Tretton37 har en kompletterande roll i deras system, där andra verktyg är centrala, men där wikin lämpar sig bättre för just kunskapsdelning inom systemutvecklargrupper på ett projektövergripande plan.

Enligt Nonaka ”triggas” organisationer att skapa ny kunskap genom att låta de anställda uttrycka det i metaforer och ge dem frihet att uttrycka även vaga idéer som får form genom dialog (Nonaka, 1994). Vi ser i våra respondenters fall att wikin inte är det enda verktyget utan både på Bool och Tretton37 är det fritt fram att använda olika verktyg. På Syncron finns det inga krav på att man som användare skriver färdigt sina tankar direkt.

Haldex använder sig inte av en wiki, men det finns ett behov av att samarbeta och dokumentera på ett bättre sätt och att få en högre grad av spårbarhet. Vi anser att Haldex mer eller mindre lyckas täcka det här behovet på ett annat sätt. Just för att åstadkomma det används alternativa lösningar såsom mailarkiv som är knutna till en position. Om man lämnar över arbetet till någon annan lämnar man även över sina .pst-filer. Vi har sett att det är många fler platser där informationen kan förekomma och vid sökandet kan det behövas gå igenom steg såsom att titta i pärmar, i en fileshare, i .pst-filer, kontakta gamla kollegor och kontakta leverantörer. Om det inte går att hitta information är man, enligt Mikael på Haldex, “tvungen att göra om allt arbete som vi redan har gjort tidigare”.

Vår slutsats blir därför att en wiki fyller behovet av att samarbeta och dokumentera till en viss nivå, men att det går att kompensera behovet av andra verktyg än wiki på andra sätt.

5.2. Wikiverktyg

På frågan om brister med wikiverktygen ansåg både Tobias och Johan att det inte fanns några större problem med deras respektive wikiverktyg. Tobias gick till och med så långt att han konstaterade att det inte fanns några som helst nackdelar med en wiki utan att de eventuella problemen som uppstod hade att göra med hur människor var "funtade". Johan däremot påpekade några mindre nackdelar med wikin, men överlag så skapade det inget obehag hos användarna, berättade han.

Hampus tyckte att problemet med wikin inte låg på dess design utan att tog för mycket tid och kraft att dokumentera på den. Å andra sidan kände han att han inte hade någon bättre lösning på hur man som användare kan dokumentera utan att det ska kosta tid och kraft.

Vår slutsats blir därför att möjliga negativa åsikter om wikin ofta beror på andra aspekter än wikins design.

På frågan om andra verktyg som används fick vi på Syncron reda på att deras wiki används till nästintill allt inom den organisationen. Det är ett gemensamt ställe för alla requirements och all intern dokumentation. Förutom en äldre Sharepoint-portal, som håller på att avvecklas, är wikin den enda plats där man kan hitta all information, vilket (alltså att ha allt på en plats) är väsentligt för en organisation vars anställda befinner sig på olika kontinenter, i olika tidszoner och som talar olika språk. Trots att det handlar om en enterprise wiki, vilken ger fler möjligheter till samarbete inom just organisationer, finns det andra verktyg som lämpar sig bättre. Till detta hör bland annat versionshantering och ärendehantering, vilka är integrerade med wikin i Syncrons fall.

Versionshantering används på Syncron, Bool och Tretton37, men inte på Haldex eftersom all systemutveckling numera är utlagd. Ärendehantering däremot används på Syncron och Haldex. Syncron använder Jira både för supportärenden och för utveckling av nya funktioner i organisationens program. Haldex har ett "gammalt casesystem", kallat HEAT, för ärendehantering. Casesystemet har dock sämre stöd för sökning, som kräver att man som användare måste känna till ärendenumret. Konsulterna på Tretton37 använder bara ärendehanteringssystem inom vissa projekt. Både Tobias på Tretton37 och Hampus på Bool berättar att de jobbar mycket agilt och försöker att visualisera uppgifter och ärenden med hjälp av en kanban och ett scrumboard. Vikten ligger på att kunna visualisera ärendehantering, vilket fungerar utmärkt i mindre systemutvecklargrupper såsom i Tobias fall (två-sex personer) och på Bool som arbetar in-house med mindre projekt (ibland så lite som en person).

Möjligheten att visualisera uppgifter på en kanban eller på ett scrumboard uppskattas både av Tobias och Hampus. Medan det används en digital tavla i projektet där Tobias sitter så föredras den fysiska motsvarigheten på Bool. Anledningen är, som Hampus uttrycker det, att det blir "extremt tydligt [och] konkret" och kan följas mycket bättre.

Vi tror att en fysisk tavla ger en känsla av gripbarhet, vilket alla andra low-technology-verktyg också ger (Alajbegovic et al., 2011).

Även Syncron har ett scrumboard på sitt kontor i Warszawa. Det tyder på att kanban och scrumboard är populära verktyg inom systemutveckling. Fast eftersom de involverade i Syncron-projekten inte sitter på en fysisk plats måste alla ärenden och uppgifter finnas i ett digitalt ärendehanteringssystem, i detta fall Jira. Det finns även situationer där wikin hade kunnat täcka kollaborations- och dokumentationsbehovet, men inte gör det på grund av avsaknaden eller ofullständigheten. Detta gäller först och främst ritningar, alltså bilder, som ska uppdateras.

Även om det är relativt lätt att lägga till bilder och skärmdumpar i wikin tar det lång tid att ladda ner dem, uppdatera dem och ladda upp dem på nytt, såsom Johan på Syncron har berättat. Därför används fortfarande Visio och andra verktyg för att skapa till exempel wireframes för gränssnitt. Visionen är att ha all information på wikin och förhoppningen är att det kommer nya och bättre funktioner i Confluence.

Även på Bool arbetar de mest med text i wikin. Alla ritningar och bilder sparas i dokumentbibliotek i Sharepoint.

Vår slutsats blir därför att användning av andra verktyg antingen beror på att de andra verktygen är mer specialiserade och vassare på sina respektive uppgifter eller att det beror på att wikin saknar stöd för just den efterfrågade funktionaliteten.

5.3. Systemutveckling

Både Tobias och Hampus ansåg att det inte spelade någon roll vilken systemutvecklingsmetod de använde, vad det gällde viljan att dokumentera inom ett projekt. Tobias ansåg att ansvaret låg på beställaren medan Hampus ville göra en uppdelning mellan intern och extern dokumentation (där det senare rör dokumentation för kunderna). Tobias gjorde ingen sådan uppdelning utan hans uppdelning handlade istället om viljan att dokumentera från kund till kund. Dock är ståndpunkten för den organisation han arbetar på, Tretton37, att allt som sker internt ska dokumenteras, men att det som sker ute hos kund (externt) är upp till kunden om det ska dokumenteras eller inte. Detta tyder på en likhet med det Hampus berättade om intern och extern dokumentation även om Tobias inte själv använde just de orden. Tobias sa dock att man inte skulle dokumentera för dokumenterandets skull utan att all dokumentation som togs fram skulle ha ett syfte, något som varken Hampus eller någon av de andra vi intervjuade konstaterade på ett explicit sätt.

Hampus förklarade vidare att vad det gällde att dokumentera för kunderna så spelade det ingen roll vilken systemutvecklingsmetod som används utan det är först när dokumentationen sker internt som systemutvecklingsmetoden har betydelse.

Det går därför att kalla den interna strukturen på Bool för ett sorts ba, då de inom organisationen möjliggör kunskapsskapande och där tron på varandra och känslan av en skyldighet att ta hand om varandra existerar, vilket skapar ett socialt kapital inom organisationen (Nonaka & Konno, 1998).

Om valet av systemutvecklingsmetod påverkade viljan av att dokumentera hade Johan inget rakt svar på. Han berättade att scrum brukar beskyllas för att inte direkt bry sig om dokumentationen, något som de flesta agila metoder beskylls för, men detta var inte något som han höll med om till hundra procent. Hans åsikt var istället att scrum minskar på mängden dokumentation för att systemutvecklingen ska ledas av systemutvecklare. Johan berättade sedan att i alla projekt han hade arbetat i var det underförstått att dokumentation var viktigt, vilket är ett typiskt exempel på outtalad kunskap (Boden et al., 2009).

Mikael skiljer sig något från de övriga respondenterna när han förklarar att hans åsikt är att det verkligen finns en koppling mellan vilken systemutvecklingsmetod som används och viljan av att dokumentera. Han gjorde istället en uppdelning på om det handlar om nyutveckling eller förändringsutveckling. Vid nyutveckling är behovet av dokumentation större än vad det är vid förändringsutveckling, förklarade han.

Mikael påpekade även att det är viktigt att anpassa systemutvecklingen till den verklighet man befinner sig i, alltså återigen en koppling till ba och socialt kapital (Nonaka & Konno, 1998).

Vår slutsats blir därför att valet av systemutvecklingsmetod inte påverkar hur villiga användarna är att dokumentera inom ett projekt utan att det istället är andra faktorer, såsom om det är intern eller extern dokumentation, som har större vikt här.

På frågan om wikin bara användes vid speciella systemutvecklingsprojekt svarade två av respondenterna, Tobias och Hampus, på väldigt liknande sätt, då de båda har projektoberoende wikiverktyg. De har även valt att inte skriva upp projektrelaterad information på dessa wikiverktyg utan de skriver istället ner generell information. Tobias sa däremot att han gärna hade skrivit in projektrelaterad information på wikin om han hade fått bestämma.

Hampus berättade att det som de skrev ner på wikin var ostrukturerat och att han tyckte att det var så en wiki skulle användas. Detta kan ses som en klar användning av designprincipen som går under namnet simpel, då den visar på att en wiki har få syntaxregler och att det därmed finns få hinder för personlig strukturering av information på den (Müller et al., 2008).

Johan förklarade att wikin används på mer eller mindre alla projekt som han är inblandad i förutom vissa enstaka kundprojekt, vilket går emot det sätt som Tobias och Hampus använder sina respektive wikiverktyg på. Johan berättade att i vissa kundprojekt har de lyckats övertyga kunden om att använda wikiverktyg när de visade upp vilka fördelar det fanns med dessa. Han sa även att han arbetar med att få alla involverade att jobba på deras wiki, som heter Confluence. Detta fungerar då en wiki är öppen, organisk, universell och delbar (Müller et al., 2008).

Johan konstaterade även att "ingenting görs förrän man skriver in detta i wikin eller i ärendehanteringssystemet", vilket motstrider Nielsens 90-9-1-regel (Nielsen, 2006, refererad i Lytras et al., 2009).

I wiki-användandet utmärker sig Syncron genom att ha wiki som ett centralt samarbetsverktyg, medan Bool och Tretton37 använder wikiverktyg som komplement till projektrelaterade system. Eftersom wikiverktyg är ett demokratiskt verktyg (Müller et al., 2008) kan de användas så som medarbetare vill.

Vår slutsats blir därför att wikiverktyg ofta används på ett projektöverskridande sätt.

6. Slutsatser

Uppsatsens syfte var att ge svar på forskningsfrågan: *Hur används wikiverktyg inom systemutveckling för att hantera kunskap?*. För att samla in empiriskt material för denna undersökning har vi som författare genomfört semi-strukturerade intervjuer med fyra olika organisationer som är verksamma i Öresundsregionen. Tre av dessa är renodlade IT-organisationer medan den fjärde är verksam inom bilindustrin, men denna organisation har också IT-projekt och är beroende av ett kunskapshanteringssystem.

För att få mer utförliga svar under intervjuerna och därmed få bättre svar på forskningsfrågan valdes det ut tre fokusområden som har varit centrala för undersökningen och som finns med i vår undersökningsmodell. Dessa fokusområden är:

- Kunskap
- Wikiverktyg
- Systemutveckling

Det sätt som ny kunskap dokumenteras på är beroende av vilken sorts kunskap (i detta fall värden) det gäller. Hårda värden, det vill säga värden som går att få ner i siffror och ord, är lätta att dokumentera på en wiki. Mjuka värden, såsom känslor och åsikter, är däremot väldigt svåra att dokumentera oavsett om en wiki finns till hjälp eller inte. Om en wiki främjar uppfångandet av ny kunskap var inget som varken bevisades eller motbevisades utifrån de svar respondenterna givit.

Wikiverktyg används på annorlunda sätt inom systemutvecklingsprojekt, där det inte är säkert att de används just för att stödja kunskapshanteringen. Wikiverktyg används på ett projektöverskridande sätt och det som väl skrivs ner på wikin är ofta generell information som inte nödvändigtvis är strukturerad på något speciellt sätt. Wikin gör det lättare för användarna att sprida information och den underlättar även vid dokumenterandet av kunskap. Wikin centraliserar även informationen, men det är inte säkert att den ersätter äldre lösningar som finns inom organisationen.

Gällande en viss systemutvecklingsmetods påverkan på viljan att dokumentera visar uppsatsens resultat att så är inte fallet. Det finns istället andra faktorer som spelar in, såsom om beställaren kräver det eller om det är intern eller extern dokumentation. När det kommer till andra verktyg som används för kollaboration istället för en wiki visar undersökningen att andra verktyg används både som ersättning för wiki, men också tillsammans med en wiki där dessa verktyg då ofta är specialanpassade för vissa situationer. En wiki däremot uppfyller oftast de samarbets- och dokumentationsbehov som en användare har och om integration med andra verktyg efterfrågas brukar det inte uppstå några större problem.

Undersökningen visar alltså på att wikin används inom systemutvecklingen i allt från kunskapsdelning på ett projektövergripande plan till att den används som en företagsportal för all kunskapshantering. Om en wiki inte existerar kompenseras detta genom andra sätt att arbeta och med andra tillgängliga verktyg.

Bilaga 1 - Ordförklaringar

Denna bilaga förklarar de akronymer som förekommer i uppsatsen och även de akronymer som togs upp under intervjuerna som genomfördes. Vissa akronymer anses vara mer eller mindre självklara och förklaras i så fall under uppsatsens gång och tas därför inte upp i denna bilaga.

- CVS:
  - Concurrent Versions System, versionshanteringssystem
  - Commercial Vehicle Systems, en division av organisationen Haldex AB i Landskrona
- DBA - Database Administrator(s), databasadministratörer
- EDI - Electronic Data Interchange, internetprotokoll
- GCM - Global Customer Master, global kunddatahantering (en produkt i Syncrons sortiment)
- GDSS - Group Decision Support System(s), beslutsstödssystem i form av kollaborationsverktyg
- GIM - Global Inventory Management, global lagerhantering (en produkt i Syncrons sortiment)
- GOM - Global Order Management, global orderhantering (en produkt i Syncrons sortiment)
- GPM - Global Price Management, global prishantering (en produkt i Syncrons sortiment)
- HEAT - Helpdesk Expert Automation Tool, ärendehanteringssystem (som används på Haldex)
- ICT - Information and Communications Technology, en beteckning för både mjukvara och hårdvara
- IPO - Initial Public Offering, börsintroduktion (det vill säga då aktier i ett aktiebolag görs tillgängliga för allmänheten)
- KMS - Knowledge Management System(s), kunskapshanteringssystem
- MDM - Master Data Management, datahantering i affärssystem
- PM - Product Management, produkthantering
- PS - Professional Services, konsulttjänster som utförs av experter
- RFC - Request for Change, ett dokument för att fånga upp och spåra förändringar i ett system (som används på Haldex)
- XP – Extremprogrammering, en agil systemutvecklingsmetod

Bilaga 2 - Intervjufrågor

I denna bilaga tas intervjufrågorna upp. Dessa intervjufrågor är kopplade till de intervjufaser som beskrivs i tabell 3.1.

Tabell 6.1: Intervjufrågorna till våra intervjuer:

Intervjudel	**Frågor**
Uppvärmning	Beskriv lite kortfattat vem du är, vad du jobbar med och hur länge du jobbat med detta. Berätta lite kortfattat om företaget, din avdelning och er produkt/tjänst.
Huvuddelen	Hur sköter ni er systemutveckling? Förklara vilka systemutvecklingsmetoder ni använder. Hur arbetar ni med kunskapshantering (till exempel skriva ner erfarenheter) inom systemutvecklingsprojekt? Beskriv kort om hur ni dokumenterar i era systemutvecklingsprojekt. Är denna dokumentation detaljerad eller generell? Sker denna dokumentation kontinuerligt?
Om wiki används	Berätta kort om er wiki. Hur länge har wikin funnits? Vilken programvara? Beskriv hur ni fångar upp kunskap inom era systemutvecklingsprojekt. Är er wiki till hjälp under denna process (på vilket sätt)? Finns det någon kunskap i era systemutvecklingsprojekt som är svår att överföra eller förmedla med hjälp av er wiki? Vilka verktyg är då bättre för den typen av kunskap? Förklara i vilka sammanhang/speciella projekt wikin används. Vilka andra verktyg utöver wikin använder ni för att samarbeta inom systemutvecklingsprojekten? Hur fungerar de ihop? Hur är det att utveckla med hjälp av en wiki jämfört med hur ni utvecklade tidigare?

	Beskriv några brister med wikin, alltså något som ni tycker kan förbättras med den. Påverkar valet av systemutvecklingsmetod ert sätt att dokumentera? Något annat/något mer ni vill ta upp?
Om wiki inte används	Hur definierar ni en wiki? Har wiki(er) använts tidigare? Om ja, vad fick er att gå från wiki(er)? Om nej, har ni planer på att implementera en wiki? Vilka samarbetsverktyg använder ni? Täcker detta/dessa program era dokumentationsbehov? Beskriv hur ni fångar upp kunskap inom era systemutvecklingsprojekt. Hur ser kunskapshanteringen ut i ert systemutvecklingsarbete? Vad gör ni för att skapa och bevara kunskap? Påverkar valet av systemutvecklingsmetod ert sätt att dokumentera? Något annat/något mer ni vill ta upp?

Bilaga 3 - Intervjuer

Intervju med Deniz på Tretton37

Den 12 april 2011

Beskriv lite kortfattat vem du är, vad du jobbar med och ungefär hur länge du har jobbat med detta.

Deniz Yildrim heter jag och är VD på Tretton37. Tidigare har jag arbetat på andra bolag, främst inom IT-konsultbranschen. Det första bolaget jag jobbade på var Varchar AB, som sysslade med systemutveckling på .NET-plattformen. Det bolaget blev sedan uppköpt av Cybercom Group, där jag arbetade som dotterbolags-VD och senare affärsområdeschef. Vid sidan om detta har jag varit involverad i ett antal andra *start-ups* och medelstora bolag, men det behöver vi inte gå in på...

Berätta lite kortfattat om företaget och er produkt/tjänst.

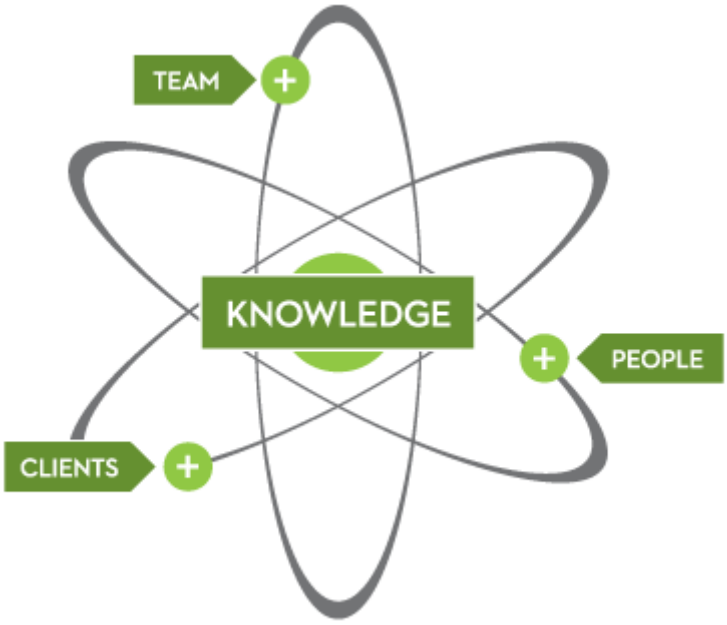
Vad det gäller Tretton37 så arbetar vi med *systemutveckling på .NET-plattformen*. Vi gör detta med hjälp av *lean* och *agila metoder*. Det är vårt erbjudande och det är det vi säljer, inget annat. Bolaget grundades i januari 2010 och vi är alltså 15 månader gamla. Idag har vi 25 stycken riktigt duktiga utvecklare i bolaget och vi kommer att fortsätta växa till runt 35 till slutet av året.

Om vi tittar på kundbilden så har vi väldigt intressanta kunder, inom väldigt olika branscher. Vi jobbar till exempel inte med kunder inom telekombranschen, som så många andra konkurrenter gör. Vi jobbar istället med industribolag runt om i regionen och dessa bolag jobbar med helt olika saker, så vi är inte beroende av bara en industri och detta gör att vi inte bryr oss om vilken industrikunskap man måste ha innan man kan börja arbeta med en kund.

Vi sitter i Lund än så länge. Vi har inte etablerat några andra kontor, men det finns i affärsplanen längre fram. Så från 2012 och framåt så kommer vi även att etablera oss i andra regioner.

Bolaget bygger enbart på *kunskap* och *kunskapsdelning*. Det handlar om att prioritera kunskap, i grund och botten, och det är tre instanser som vilar på kunskap och dessa är:

1. **People**: de som arbetar inom bolaget eller de som ingår i *communityn*.
2. **Team**: de individer som arbetar inom bolaget.
3. **Clients**: de som egentligen betalar våra löner.



Figur 6.1: Tretton37:s kärninstanser. Tagen från Tretton37.com (2011-04-13).

Finns inte kunskap i grund och botten så finns vi inte. Det är den ena tillgången som finns i bolaget. Därför är det väldigt viktigt att vi förvaltar den kunskapen på ett bra sätt. När vi tänker på de instanser som drar nytta av kunskapen så går det inte att prioritera den ena eller den andra. Om vi prioriterar konsulten så kommer kunden inte att vara nöjd i slutändan just därför att konsulten inte kommer att representera vårt bolag med högsta möjliga potential och engagemang. Det är därför viktigt att vi kan få våra konsulter att inse att de är minst lika viktiga som våra kunder. Våra konkurrenter säger "nej, kunden är i fokus", men visst, hos oss är kunden fortfarande i fokus, men konsulten ska också vara i fokus. Det är viktigt att individen känner att han/hon utvecklas som utvecklare och att han/hon växer tillsammans med bolaget.

Hur sköter ni er systemutveckling?

Vi är konsulter och vi anpassar oss efter kundens behov, men samtidigt så vet vi att vill man ha ett bra resultat så bör man använda en viss metodik. Vi vet att de gamla modellerna, typ vattenfallsmodellen, inte fungerar lika smidigt eller ger lika bra resultat som de agila metoderna. *Vi förespråkar därför gärna agil metodik*. Är kunden villig att applicera detta i sina miljöer så gör vi detta, men några gånger så är detta inte möjligt för att det är en stor förändring i deras organisation också. Vi tar då in en specialist som talar om riskerna som finns med denna utvecklingsmetodik och vi försöker förklara att man kan göra detta på ett bättre sätt. Man tar det eller inte. Det är kundens beslut. Vi har lyckats övertyga två av våra större kunder om detta och få dem att inse att detta producerar bättre resultat. Så länge slutkunden blir nöjd så känner vi att det är viktigt att använda den metodiken på grund av det. Det är alltså så vi arbetar hos kunden.

När det gäller våra konsulter så vill vi att de delar samma värderingar som de andra anställda, det vill säga att de visar intresse för samma typ av utveckling. Vi känner att det är viktigt att rita en utvecklingskarta för att visa att så här ska det gå till; vi vill att du ska göra detta, vi vill att du ska fokusera på detta etc.

Vi har väldigt många olika kunskapsdelningsforum inom företaget. Detta innebär alltifrån interna seminarier till utbildning av agil metodik. Så hela idén bygger på att de som kommer in som nya i bolaget ska bli upplärda av de som har arbetat med dessa utvecklingsprojekt tidigare.

Men arbetar ni efter några fördefinierade metoder, såsom scrum, eller har ni utvecklat er egen metod? Eller är det bara så att det ska vara agilt?

Prio ett är att det ska vara agilt. För varje systemutvecklingsprojekt är speciellt och varje kund har speciella behov. I många fall kan man applicera delar av scrum eller delar av kanban eller delar av XP... Däremot att ta oss fram för givet inom alla organisationer hade varit orealistiskt. *Så det vi utgår ifrån egentligen är principen för agil utveckling och sedan så anpassar vi oss efter kundens behov*. Vi har alltså inte en egenutvecklad modell, då den inte hade fungerat i alla olika sorters organisationer.

Alla organisationer har sina egna behov då alla organisationer är unika, precis som människor, och därför måste man ha respekt för det också.

Hur arbetar ni med kunskapshantering inom era systemutvecklingsprojekt?

Vi har en grund i vårt företag; kunskap. Sedan har vi tre instanser som vilar på denna grund; *people*, *team* och *clients*. Jag sa innan att det inte går att prioritera det ena över det andra. Våra konsulter ska behandlas på samma sätt som vi behandlar våra kunder. Först och främst så när de ska allokeras till ett projekt så ska de veta vad detta innebär, vad det är för typ av projekt och om de kan bidra med ett högt värde inom detta projekt. Vi kastar aldrig ut våra konsulter i projekt och säger "men gå dit och lös det". Det funkar inte inom vår verksamhet.

Så, eftersom kunskap är vår grund i företaget, så måste man vårda denna. Hur kan man då vårda kunskap? Det måste finnas en *knowledge sharing process*, som vi kallar det för, alltså en kunskapsdelningsprocess mellan dessa instanser. Jag brukar visa det så här [Deniz ritar figur B.1 på en whiteboard]. Ju mer kunskap vi har, desto bättre idéer har vi och desto bättre individer vi har desto bättre team får vi. Desto bättre team vi har, desto bättre resultat blir det, vilket leder till en bättre leverans. Då blir kunden nöjd och kommer tillbaka till oss när han/hon behöver nya saker. Detta innebär att vi kan *boosta* vår kunskap med nya erfarenheter, vi kommer att lära oss nya saker, vi kommer ta nya utmaningar, vilket gör att vi boostar kunskapen på det sättet.

Vi har olika diskussionsforum med communityn. Vi bjuder in intressanta profiler från olika bolag, till exempel konsultbolag, produktutvecklingsbolag, kundbolag eller leverantörsbolag. Det spelar ingen roll. Vi samordnar även olika typer av *code nights*, där vi hittar på roliga saker såsom *robot fight nights*, där vi vill ha in duktiga utvecklare som kommer hit och leker med oss. Där ska du lära oss något som du kan bättre än oss, men samtidigt så ska du lära dig av oss. Vi delar därför kunskapen med communityn både ut och in.

Ett annat diskussionsforum vi har heter *software craftmanship*, där vi försöker se om vad det innebär för oss och vad det innebär för communityn. Tolkar vi det på samma sätt? Vissa tycker att craftmanship är alldeles för kostsamt då man fokuserar för mycket på kvaliteten. Andra tycker att det är värt tiden och pengarna man investerar i det. Vi för därför filosofiska diskussioner om vad craftmanship är och vad det kan göra. Vi samlar därför aktörer från hela regionen för att diskutera om detta.

Vi ska även dela kunskapen inom bolaget. Som nämnt innan så har vi code nights, varje onsdag faktiskt, där vi testar saker *hands-on*. Vi har även *techradar*, som är ett frukostseminarie som vi håller varje fredagsmorgon. Där tar vi upp vilka trender som verkar intressanta, vilka teknologier som verkar intressanta, som vi kanske kan arbeta med i framtiden. Vi har även *knowledge sharing moments* som hålls en gång i månaden. Det är ett internt seminarie, där en av våra anställda tar fram en utmaning, alltså något han/hon är intresserad av och där han/hon lär oss om dessa tekniker. Slutligen har vi *knowledge sharing weekends* där vi åker bort till en stuga över en helg. Där har vi både interna och externa talare och där vi tar upp många olika frågor och ser hur vi kan testa detta hands-on. Så en av dagarna är det bara konferenser/seminarier medan det enbart är hands-on-övningar den andra dagen.

När vi ska dela kunskap med vår klient så tycker vi att seminarier är *old news*, då alla sysslar med detta. Istället anordnar vi *code camps* där våra klienter tar hit sina utvecklare och där vi testar nya saker tillsammans. Vi delar kunskap hela tiden, alltså in och ut. Det är så vi ser till att kunskap alltid är *up-to-date*. Vi har även något vi kallar för *knowledge committee* där fem av våra konsulter ingår. Jag är inte en av dem. Dessa fem är med och vill driva bolaget framåt. De ser även till att dessa aktiviteter verkligen genomförs. De är de som sätter igång processen. Allt går igenom dem. De kollar även på vår budget för att se så att dessa aktiviteter är möjliga. Alla är därmed involverade i utvecklingen av bolaget. Alla är involverade i beslutsfattandet. Detta är något vi är väldigt stolta över. *Detta är hur vi driver och underhåller kunskap inom bolaget*.

Två andra saker som är viktiga att veta är bolagets *vision* och *mission statement*. *Vår vision är att vara det mest beundrade bolaget i regionen*. Med att vara beundrad menar vi att vi ska fokusera på rätt saker så att våra kunder och potentiella partners ska beundra oss. Vi fokuserar därför på *kompetensutveckling*, *gemenskap* och våra *värderingar.*

Vår mission statement är beskrivning på hur vi ska uppnå vår vision, men även en påminnelse på varför ett bolag finns. Vårt mission statement är "Creating value by sharing knowledge". Värde är ett ord som betyder olika saker för olika människor. Vi vill dock uppnå värde genom att dela kunskap. Genom att applicera detta i praktiken så ska vi bli det mest beundrade bolaget i regionen.

"What you see is who we are". Vi är ett transparent bolag. Vi har inga hemligheter, varken inne i bolaget eller utanför. Det finns naturligtvis vissa saker som man inte brukar tala om, till exempel, hur vi arbetar med våra kunder. Vi vill inte att konkurrenter ska veta detta, men sådant här är väldigt viktigt att tala om.

För det här är ett bevis på hur vi tänker, vilka värderingar vi har.

Är det så att man kan dela med sig av kunskap utan att förlora konkurrenskraft?

Ja, det kan man för det finns ju andra saker i detta då. Genom att göra det på det sättet får man uppskattning i communityn, alltså genom våra insatser i communityn. Vi *tar* inte bara, vi *ger* också tillbaka. Det är väldigt viktigt att vi läser bloggar, att vi är med på vissa seminarier och i vissa *user group*-aktiviteter, men det är även viktigt att ge någonting tillbaka. Om våra kunder betalar för våra tjänster är det viktigt att ge dem någonting tillbaka och det handlar inte bara om att hitta på sådana där gratisluncher eller liknande, utan att faktiskt göra något konkret så att de kan bli bättre på det de gör. Detta är väldigt viktigt för oss.

Det är lite open source-tänk, eller hur?

Jo, absolut. Vi är rätt... jo, i och för sig, Microsoft har också blivit lite öppnare. Förr i tiden när någon hackade deras plattformar så fick de en stämningsansökan, medan nu så blir de nästan hyllade.

Enligt det du har sagt och ritat på tavlan så har ni mycket mun-till-mun-kunskapsdelning på företaget. Det vi undrar är om ni skriver ner kunskap också?

Absolut. Vi dokumenterar de flesta av våra *events*. Vi för protokoll på många av dessa events. De eventsen där det är diskussionsforum, så dokumenterar vi. Vi identifierar så kallade *action points* och vi delegerar de här action pointsen till olika individer, alltså till de som har möjlighet och kompetens för att genomföra dem.

Kan du förklara vad du menar med action points?

Det är "att göra"; to do. När det gäller de här code nights till exempel, så är resultaten dokumenterade och finns på våra servrar. På vår senaste code night så ledde det till en hjälpsajt för Japan, som nyligen har drabbats av jordbävningar. Vi ville skapa en sajt med information på hur det går medarbetet där och hur man kan donera pengar och hjälpa Röda korset och sådant. Så vi har faktiskt tillverkat en sajt som kommer att släppas nästa vecka. Den kallas för *Ninja-aid.com*. Det är en *mashup-sajt* som innehåller statistik, en länk till Röda korset och hur man kan hjälpa etc.

Det är ett kul initiativ och det är också ett sätt att vi delar vår kunskap för att hjälpa till de som behöver. Det här har säkert kostat oss hur många timmar som helst, men vi ser detta som ett bidrag till communityn. Vi har sagt "okej, det kostar oss pengar, men människor som behöver detta kommer att dra nytta av det".

På det sättet blir man beundrad, uppskattad och får respekt, tycker jag.

Så om vi har förstått det rätt så är Tretton37 aktiva på sociala medier och ni delar med er av kunskapen utanför företaget och skapar på det sättet kunskap inom företaget.

Kunskap är ett väldigt dynamiskt koncept. Om man inte uppdaterar sig hela tiden så blir det snabbt old news och detta gäller speciellt i vår bransch. Just det drivet som finns i communityn är helt otrolig. Oavsett hur mycket man dokumenterar så blir det old news om några veckor eller om några månader, så det är lika bra att satsa framåt hela tiden.

Det är dock viktigt att dokumentera det på ett annat sätt också och åtminstone att dokumentera processen kring det. Alla våra medarbetare vet hur vi följer upp dessa steg. Informationen mot våra anställda är väldigt viktigt. Vi vill också att de ska *committa* sig till det här då. Om man inte vet hur man sköter processen eller hur den funkar så är det helt omöjligt att committa sig till den typen av aktiviteter. Det är därför viktigt för oss att visa processen och visa vårt mål, det vill säga huvudmålet.

Hur visar ni detta då?

På väldigt olika sätt. Vi har etablerade kommunikationsforum, alltifrån veckomail, månadsmöten, olika diskussionsforum till workshops och sådant som finns i bolaget. Som VD är det min främsta uppgift att motivera folk att committa sig till visionen.

Finns det något intranät, som ni kan samla informationen på?

Det finns ett arbete kring det, ja. Just nu finns det lite spridda delar såsom dokumenthantering, lite olika projektplatser lite hit och dit. Det vi håller på att göra just nu är att samla allting på ett och samma ställe.

Vad som gäller dokumentation så vill vi också fråga wikiverktyg, vår inriktning. Använder ni en wiki på ert företag?

Jag skulle faktiskt vilja hänvisa er till en annan kollega när det gäller wiki-frågan. Vi har lite olika typer av *forum*, som vi använder vad som gäller systemutveckling, bland annat GitHub, som är något liknande konceptverk. Det finns en wiki där också. Vad det gäller hur wiki används, hur ofta den används och hur man dokumenterar saker i den, så kan jag inte svara operativt, därför att jag är själv inte involverad i systemutveckling på det sättet. Vill ni ha mer detaljerad information, så får ni prata med en av mina kollegor så att ni får bättre information om hur det fungerar i praktik.

Tack. Det ska vi göra.

[...]

Det är viktigt att aspirera och ge mening till det man gör. Utan det skulle man bara tillverka kod utan någon större mening och sammanhang. Det är viktigt att ge syfte och sätta det i sammanhang.

Oftast brukar man inte berätta om vad man gör i bolaget internt, men vi är ett transparent bolag. Vi vill att alla ska känna sig att man vet exakt vad som pågår på Tretton37. Det är det sättet som vi skapar förtroendet på marknaden också. Man fattar att vi har tydliga visioner, man fattar att vi har ett mål, att vi går mot ett genomtänkt mål. Inte bara att vi ute efter snabb vinst. Vi hade kunnat rekrytera många. Att bli hundra personer snabbt är absolut inte målet. När vi rekryterar folk är vi extremt noggranna med att folk vi anställer delar våra värderingar, alltså mjuka värderingar, och värderingar inom systemutveckling, kvaliteten man tillverkar och, så klart, hur duktig man är på den tekniska plattformen och hur duktig man är på sociala plattformen. Det är väldigt viktigt för oss. Hade vi velat växa, utan kontroll alltså, hade vi blivit 50 personer snabbt. Vi har så många uppdrag, så många kunder som bara vill köpa.

Vi hade kunnat anställa vem som helst. Resultaten är kortsiktigt då, med tankesättet "okej, vi kan tjäna skitmycket pengar i år, men om ett år kommer man tycka att vi levererar skit och det inte är kvaliteten vi levererar". Så det är väldigt viktigt för oss att ha en plan och det är viktigt att kommunicera detta utåt. Alla ser faktiskt vad vi håller på med. Det är ett sätt att visa öppenheten. [...] Jag är inte kodare på det sättet, men jag har kompetens att driva ett sådant kunskapsbolag där det är viktigt att visa öppenheten.

Tretton37 erbjuder konsulttjänster inom .NET. Varför kallar ni er för ninjas då?

Vi tycker att ordet konsult, eller specialist som vissa kallar sig, inte speglar det värdet som det borde speglas i dagens konsultmarknad. Alla profiler kallar sig för konsulter, alltså alla uthyrbara resurser är konsulter. Är du truckförare? Är du städare? Det spelar ingen roll. Du är konsult och blir uthyrd på timbasis. Vi är så pass rebelliska på den här marknaden, så vi tänkte "okej, de kallar sig för konsulter så vi vill inte kalla sig för konsulter, men vad ska vi kalla oss för då?". Vi valde att hitta på en benämning som är så pass *bisarr* och *pirataktig* att vi ska utmana marknaden lite grann. "Okej, ni kallar er för konsulter, då kallar vi oss för ninjas så får vi se vem som är duktigast". Så egentligen spelar det ingen roll vad du kallar dig för.

Det viktigaste är vad du visar för resultat. I början tyckte vissa att vi var oseriösa för att vi kallade oss för ninjas, men faktum är att i Skåne levererar vi, enligt mig, den bästa kvaliteten och vi kallar oss inte för konsulter eller specialister. Vi är ninjas.

Tack så mycket för intervjun. Det var snällt att du ville ställa upp.

Det var bara roligt att kunna hjälpa till.

Kompletterande intervju med Tobias på Tretton37

Den 13 april 2011

Tack att du ville ställa upp på en intervju. Vi pratade med Deniz igår som berättade om Tretton37, om företagets visioner och arbetet med kunskap. Vi vill därför gärna ställa mer praktiska frågor till dig:

Använder ni någon sorts wiki här på Tretton37?

Jag använder det i alla fall på uppdraget jag sitter på. Både för projekt, men även som redskap inom gruppen, då projektöverskridande för att dokumentera arbetssätt och andra rutiner som finns.

Hur dokumenteras det i ditt projekt?

Wikin används för intern dokumentation inom utvecklingsgruppen. Projekten som sådana har andra dokumentationskrav: det är *hederliga* [skrattar] Word-dokument enligt vissa format.

Du menar dokumentation för slutanvändare?

Ja, precis. För andra intressenter i projektet. Det kan vara *technical management* som hanterar *maintenance*, alltså servrar och den biten, men det kan även vara de som sköter support, som tar emot supportärenden så att de ska kunna felsöka. Det kan vara dokument riktade till beställaren i form av förklaringar hur produkten fungerar. Wikin använder vi, som sagt, enbart inom projektgruppen för att dokumentera.

Så den är bara tillgänglig inom er utvecklingsgrupp, eller?

Nja, den är tillgänglig för alla, men den är ämnad för oss. Huvudsyftet med denna wiki är för oss utvecklare.

Hur länge har wikin funnits?

Där vi sitter nu är det sen vi fått Sharepoint, som är plattformen just nu.

Är det wikin i Sharepoint som ni använder eller är det vanliga sidor i Sharepoint som ni använder som wiki?

Det är den riktiga wikin på Sharepoint som vi använder. Jag har använt MediaWiki också, fast i tidigare projekt, men den här är Sharepoints wiki.

Funkar den wikin bra, tycker du?

För vårt syfte funkar den bra.

Hur funkar det att fånga upp kunskap inom ditt projekt med hjälp av wiki?

Just uppfångning vet jag inte. På sätt och vis gör vi ju det. Saker och ting som vi känner går på rutin för vi in i wikin, för det är först och främst det som ska komma in där. Jag vet inte om det är mediet wikin som sådant som gör det eller om det är att vi har ett intresse av det. Sättet man arbetar med wikin förenklar ju det. Det är enklare än att öppna Word-dokument och redigera. Jag vet inte om det är själva wikin som driver detta utan behovet i projektet.

Hur många är ni som deltar i projektet?

Projektstorleken varierar. Vi har ofta wikiverktyg för aktiviteter som är projektöverskridande. 1-2 upp till 6 personer, kanske. Så det är inte bara ett projekt, utan det är många olika.

Finns det andra verktyg som används förutom wikiverktyg inom projekten?

Ja, det är vanliga Word-dokument. Jag nämnde inte oss själva, men även vi är intressenter i projekten. Vi har dokument med standardmallar för andra som ska arbeta med systemet.

Vilken systemutvecklingsmetod kör ni?

Det varierar lite. Jag skulle vilja säga scrum, men det är nog egentligen ScrumBut [skrattar], som vi brukar kalla det. Det är snarare så att vi använder oss av agila principer i de projekten vi kör, men om man tittar på projekten från start till slut, så är vi inte alltid där. Så det är mer vattenfall i visst skede. Det är först några iterationer då vi börjar köra agilt.

Hur är det med andra verktyg såsom post-it-lappar och whiteboard? Använder ni sådant?

Ja, vi använder både post-it-lappar och whiteboard i våra projekt eller scrumboard som det heter inom scrum. Vi har även provat att köra ett digitalt scrumboard, så det är väl inget som hindrar det. Om det är vikt i information i post-it-form, så att det måste dokumenteras, finns det elektroniskt också. Ofta föds det som finns i scrumboard från en *backlog* som i sin tur finns elektroniskt, så att det blir en arbetskopia som du sedan kan utveckla under iterationen. När iterationen är slut så är all dokumentation i kod i stället. Då är det inte så intressant att ha det kvar elektroniskt.

Vad är det för verktyg för elektronisk scrumboard ni använder?

Vi kör Team Foundation Server och sen har vi en klient för det. Telerik heter företaget som utvecklar den klienten för att visualisera scrumboard.

Är det några speciella projekt eller några speciella sammanhang där wikin används?

Det är inte direkt för projekten, utan det är snarare för oss utvecklare som jobbar där wikin används. Det är mycket att beskriva komponenterna vi använder, informationen för hur de fungerar och installationsprocesserna. Det är mer åt det hållet än dokumentation kring själva projekten. Den främsta anledningen till detta är att intressenterna vill ha det i annat format än wiki. Hade de önskat det i wikiformat, så hade vi kört på det i stället. Det är i och för sig ingenting vi har frågat dem om, men...

Hur tycker du är att utveckla med hjälp av wiki jämfört med tidigare, alltså utan wiki?

Fördelen med wiki är att det är lättillgängligt och det är ett lätt medium att redigera och arbeta i och vidareutveckla dokument eller dokumentation. Du har betydligt större frihet att skapa den struktur du vill ha. Så det är väl det som är fördelen med att utveckla med en wiki och att inga begränsningar finns i hur du utformar dokumentet ifrån början, så du kan sätta de standarder du vill. Har du ett verktyg för att dokumentera, har du vissa begränsningar.

Har det funnits problem att övertyga folk att använda wiki?

Problemet är snarare att få folk att dokumentera. Alltså dokumentation generellt sett, inte bara i wikiverktygen. Återigen, wikin är lättillgänglig och lätt att arbeta i.

Men det har alltså inte varit några problem att övertyga folk att skriva ner?

Jo, det kan absolut vara. Jag kan själv ha problem att skriva ner saker, men det handlar snarare om hur vi är funtade som människor. Där är det återigen enkelt att arbeta i wikin och därför finns det inte så mycket att klaga på.

[...]

En sak jag tänkt på med MediaWiki var att det krävdes en viss kunskap hur det fungerar, om du vill kunna formatera. Det är så fall det som är begränsningen, men samtidigt är det syftet med wikin att vara enkel och inte ha för avancerade saker.

I dina projekt, har du använt andra wikiverktyg förutom MediaWiki och Sharepoints wiki?

Inte vad jag kommer på. Bara Sharepoints wiki och MediaWiki. Finns det många andra?

Det finns Confluence, till exempel, vilken är en enterprise wiki som bland annat integrerar issue tracking system. Sedan finns det TWiki som är skapad just för systemutvecklingsprojekt.

Aha, okej.

Tror du att valet av systemutvecklingsmetod påverkar sättet att dokumentera?

Där jag jobbar är ambitionen att dokumentera så lite som möjligt. Med det menar jag inte något negativt. Den dokumentation du ska ta fram ska ha ett syfte. Det finns så många fall där man har dokumenterat för dokumenterandets skull och där processen kring dokumentering varit olidlig. Jag vet inte om jag har känt av att det har med valet av själva metoden att göra. Det är snarare vilka krav som finns hos beställaren, oavsett metod.

Finns det andra fördelar med wikiverktyg för systemutvecklingsprojekt?

Om du sitter och gör en iteration och om det blir fel i iterationen, kan du se vem som har ändrat i wiki-dokumentationen. Du kan se historiken. Det hade varit svårt att uppdatera ett Word-dokument och det är omöjligt att se historik över ändringar. Så när nästa person kommer in blir det också fel, fastän det har blivit dokumenterat. Det är smidigt med wiki. Dokument i Sharepoint är också ganska smidigt, men fördelen med wiki där är just länkningsmöjligheterna och att du slipper uppladdningsprocessen.

I Word-dokumentation brukar det finnas tabeller med information om vem som har upprättat, vilken revision det är, vem som har uppdaterat och när och vad. Detta kan vi tänka oss tar tid att fylla i.

Saken är att hade du gjort en *diff* på en wiki-sida, så hade du direkt sett alla aktuella ändringar, så att man inte behöver fylla i sådana tabeller för hand. Det är styrka i det också.

Tack så mycket för intervjun. Det var snällt att du ville ställa upp.

Inga problem.

Intervju med Johan på Syncron

Den 14 april 2011

Beskriv lite kortfattat vem du är, vad du jobbar med och ungefär hur länge du har jobbat med detta.

På Syncron har jag jobbat sedan 2001, minus ett halvår då jag var ute och såg mig omkring. Jag började jobba som Java-utvecklare och jag har haft varierande arbetsuppgifter, däribland lite projektledning, men väldigt mycket jobba som intern konsult när vi hjälper kunden med konfigurerandet. Numera jobbar jag inom vår produktutveckling, alltså PM (Product Management).

Vår avdelning är uppdelad i två delar. En Business Analyst-del som sitter här i Sverige och det är min roll; Business Analyst. Jag ansvarar för våra MDM-lösningar (Master Data Management), typ artikelmaster och kundmaster, men så mycket jag hinner är jag även ansvarig för vår GOM-produkt (Global Order Management).

Vår andra del av produktutvecklingen sitter nere i Polen, en 20-30 personer som utvecklar. Eller, alla utvecklar inte utan det jobbas även med test och support och några jobbar som konsulter. Vi utvecklar inte så mycket i Sverige utan vi gör det på vårt kontor i Warszawa.

Du kom in till Syncron när det var i sin vagga.

Ja, Syncron, eller Sync som det hette när jag började, startade 2000 eller rättare sagt så skapades företaget nog 1999, men det var under 2000 som det tog fart. Jag kom in under hösten 2000 när jag skrev min magisteruppsats och jag blev sedan anställd i början av 2001.

Vi var två bolag som gick ihop 2004, tror jag det var. Sync, som låg här nere i Malmö, och Syncron, som fanns i Stockholm. Nu är vi cirka 120 personer i bolaget, som är spridda i Polen, Japan, USA, England, Tyskland och Sverige.

Du nämnde GOM och MDM. Är det någon av dessa du sysslar mest med och hur länge har du sysslat med precis detta?

Det är MDM, absolut. Vi har några olika produkter. Här nere i Malmö är det de produkter som rör GOM som vi sysslar med. Inom MDM, som jag sysslar med, har vi artikel-, kund- och leverantörsmaster. Sedan har vi något vi kallar för GPM (Global Price Management), som handlar om hur priser ska sättas, för det är en hel vetenskap i sig. Uppe i Stockholm, där vi utvecklar vår Supply Chain Planning-produkt, går detta under namnet GIM (Global Inventory Management), som berör hur man räknar ut hur mycket man ska ha i lager och även det är en hiskelig vetenskap i sig.

Jag har sysslat med MDM till och från rätt så länge, även innan vi ens kallade det för MDM. Vi byggde ett register för kunder i GOM innan, men vi gjorde sedan det till egna produkter eftersom många kunder krävde detta. Så, jag har jobbat med det i stort sett hela tiden jag har varit här. Mer eller mindre aktivt i åtminstone tre år.

Kan du förklara lite mer vad det innebär att jobba som Business Analyst?

Ja då. Det är två olika roller, beroende på om man jobbar med produktutveckling som konsult inom PS (Professional Services). Inom produktutvecklingen idag så handlar det om att fånga upp kundkraven. Detta gör vi inte direkt, men vi tar del av kundkraven som vi får ifrån de som jobbar mer direkt med kunderna. Vi ser även på våra konkurrenter och ser vad de jobbar med.

Vi ser då på vad vi måste förändra med produkten för att hålla oss i framkant med denna produkt. Som Business Analyst så är det min uppgift att definiera och specificera kraven så att vi kan ge dem till en utvecklare som senare gör implementeringen. Som konsult jobbar man mer med att fånga processen som kunden har och specificera hur vi ska implementera vår produkt hos kunden.

Det är nämligen produktutveckling, alltså Java-kodning (den enda typ av kodning vi sysslar med här), som man specificerar medan det andra är hur vi konfigurerar vår produkt.

Jobbar du som båda rollerna?

Nej, jag håller mig bara till produktutvecklingen, alltså att specificera för Java-utvecklingen. Tidigare så höll jag på med det andra. Jag har bytt roll lite inom företaget.

När du har fångat kraven från kunderna, hur vidarebefordrar du sedan dessa till utvecklarna?

Det beror lite på hur stor sak det är. Är det en liten grej så sker det mestadels genom att specificera på vår wiki eller att genom mail eller telefon meddela vad de ska göra. Är det en större sak som ska utvecklas så vill vi träffas, då oftast i Polen, för att gå igenom vad som ska göras. Uppföljningen sker oftast via mail, telefon eller Skype.

Är du ofta i Polen?

Nej, inte så ofta. Jag tycker inte om att resa för mycket inom jobbet, så det ska vara lagom. Har man barn hemma så vill man inte vara borta för mycket heller.

Vilka systemutvecklingsmetoder använder ni?

Vi jobbar rätt så mycket med scrum. Det började vi med sedan något år tillbaka. Sedan finns det ju lika många varianter av scrum som det finns folk som använder det, men grundkoncepten med en *backlog*, *daily stand-ups* och ett *scrumboard*. Jag tycker det funkar bra. Vi använder även det på våra implementationsprojekt när vi är hos en kund.

Du kom in på kunskapshantering innan. Du nämnde wiki innan till exempel...

Ja, det är framförallt det vi har. Då är det *Confluence* vi har. Vi hade *Inside*, vårt interna system på Sharepoint, innan. Fast nu har vi en gammal version av Sharepoint som vi inte underhåller längre, så den fasas ut. Där finns viss information kvar och den används aktivt inom vissa projekt, men vi tycker att den är mycket sämre än vad Confluence är. Det kan ju delvis bero på att det är en gammal version vi jobbar med, men Confluence är mycket smidigare.

Beskriv lite kort hur ni dokumenterar. Är det detaljerat eller generellt?

Det är både och. Det beror på när och vem det är som gör det. Vi börjar ofta mer generellt och sedan går vi ner på djupet. Vi använder oss av ärendehanteringssystemet *Jira*, som också är utvecklat av Atlassian, precis som Confluence. De jobbar rätt bra ihop och man känner igen sig. Det är just med Jira som vi brukar fånga nya krav, då både buggar och utvecklingsmöjligheter. Där blir informationen rätt generell och kortfattad. Om man sedan går in på djupet, som jag som Business Analyst gör, så får man gå ner på den nivån så att en utvecklare kan förstå vad det är man vill få fram. Då är det både text och *wireframes* för att bygga upp hur ett webb-GUI ska se ut. Sedan har vi även kopplat en egengjord *template* i Excel till vår wireframe; vilka fält som ska användas, hur dessa valideras, vilken text som ska stå i vilka rutor och sådant. Wireframsen gör vi i Visio och jag personligen brinner för att lägga in så mycket som möjligt i Confluence. Visio och Excel är ganska lätta att koppla tillsammans med Sharepoint, eftersom de alla är utvecklade av Microsoft.

Confluence kommer med bättre plug-ins för att koppla ihop Office-dokument, men jag vill ha in så mycket som möjligt i Confluence, då det är *bra, snabbt och folk använder det*. Ligger dokument kopplat på en sida så klickar folk inte på det, för då måste du öppna Word för att se dokumentet och det tar en massa tid och är jobbigt. *Framförallt utvecklare är lata, då det ligger i deras natur.* Detta är oftast väldigt bra, på sätt och vis, fram tills det blir dags att dokumentera. Då ska de inte vara allt för lata, men de tycker jag faktiskt att de är riktigt duktiga på. *Får du inte upp det direkt på sidan så läser du inte det*. Du tröttnar på att hålla på att öppna dokument hela tiden. I Confluence får du upp en text direkt på sidan.

Men om det är en wireframe...?

Ja, precis. Där är mitt problem, för där är det fortfarande Visio vi kör med. I Confluence finns där något som heter *Gliffy*, ett tillägg som gör att du kan göra dina bilder direkt i Confluence. Det är något jag tror på väldigt mycket, nämligen att du ska ha det direkt. Man ska inte behöva öppna flera olika dokument för att se det man vill se. Jag tror att det är väldigt viktigt för att folk ska läsa det. Det ska även underhållas. Är det jobbigt att underhållas så underhålls det inte. Det gäller då om dokumentationen sker kontinuerligt.

Såsom vi jobbar så släpper vi olika *releaser* av våra produkter. Vi jobbar då ungefär på ett sätt som man gör i en Java-miljö, där vi jobbar i *HEAD* och där vi har dokumentation som uppdateras.

När vi sedan ska släppa vår produkt så har vi den informationen plus all information på Confluence. Då gör vi en kopia av detta och märker upp den.

Då har vi ett *snapshot* på hur dokumentationen var vid precis den releasen. Sedan när vi ska släppa en ny version så tar vi ett nytt snapshot. Det är en hel kopia på allt som finns där.

Hur fungerar det rent tekniskt att ta ett snapshot?

Det är bara att göra en kopia på allt som ligger på Confluence, i det *spacet*. Jag kan visa [visar på skärmen]. Dokumentationen förändras och har vi bara ett space, som det kallas i Confluence, för den här produkten så kommer vi inte att kunna se hur vi definierar upp hur *specarna* såg ut vid en viss version. För att slippa lägga till information om vilken version det var så gör vi en kopia. Jag kan visa på den vi håller på med just nu. När vi är klara med all utveckling och dokumentation så gör vi en kopia.

Jaha, så det finns en möjlighet att göra en kopia direkt i Confluence?

Absolut. Då går man in på *Space Admin* och sedan väljer man *Copy space* och då har man en exakt kopia av det jag har just nu. Säg att det kommer några buggar på den aktuella versionen, så uppdaterar man just det spacet som med de här förändringarna. Samma tanke som med koden i Java, att du jobbar på HEAD, men aldrig utvecklar på HEAD, men sen när du släpper en ny version så *branchar* du av eller gör en ny version i ditt versionshanteringssystem, typ i Subversion eller vad det nu är du jobbar i. Samma tanke har vi då på *copies*.

Det är väldigt bra att man kan göra det. Det är lättare än att hålla på och kopiera filer.

Ja, det är lätt och smidigt. Jag kan visa var vi har våra olika specifikationer som hör hit [visar på skärmen]. Här försöker vi beskriva vad den här gör. Det är lite tunt här. Den var påbörjad i Sharepoint, innan den gick över hit och den är lite kort, som sagt. Om vi börjar med olika nivåer på specar.

Vi har också en *Software Architect*. Så vi har rena kodare och Software Architects som ska designa system ifrån *requirements* ner till specifikationer som används av utvecklarna. Så det här är högsta nivån. Då behöver man egentligen ingen kunskap om utveckling eller programmering. Här har vi fortfarande hantering av dokument. Vi har tyvärr inte kommit i mål än. Min vision är att bli av med dokumenten.

Här har vi wireframes som vi pratat om i Visio och även i Excel-arken som beskriver hur fälten styrs. Om vi går ner ännu längre, så har vi *webb-interfacet*. Det är det som görs i Visio, men har vi då lagt ut skärmdumpar och bilder av det så att man kan se dem direkt i Confluence. Då slipper jag öppna upp Visio, men denna är statisk, alltså det är en bild och går inte *editera*. Då vill jag ta fram möjligheten att kunna editera den direkt i Confluence, så att man slipper ta skärmdumpar, för att det kommer inte underhållas och att det är tidsödande.

Ungefär hur länge har ni använt Confluence?

I tre år ungefär, tror jag, och då var det PM, alltså produktutvecklingen, och ett av kundprojekten som började att använda Confluence. Hos en av våra kunder behövde vi det för att vi helt enkelt inte hade något bra verktyg för att hålla koll på alla de förändringar vi gjorde. De hade sitt ärendehanteringssystem, men det var mer kopplat till vår kund och de som ägde det. Det hade inte spritt heller att vi gjorde lite förändringar här och där plus att man inte heller alltid vill skriva i kundens system exakt hur man gör. Det kan vara känsligt och man vill också skriva saker internt. Därför tog vi fram ett system och det var Confluence vi fastnade för. Vi tittade på lite olika och Confluence kändes mest smidigt.

Vi försöker uppdatera Confluence hela tiden, både vad det gäller PM och på kundprojekten. Produktutvecklingen har vuxit väldigt mycket. Det finns faktiskt väldigt mycket dokumentation här idag. Allt från installation, *release notes*, utbildning och instruktioner, sådant som kunderna bör veta och sedan då *specar* hur vi ska utveckla produkten.

Dokumentationen för kunden då? Kan man skapa den i Confluence också?

Ja, det kan man göra. Vi har inte varit så duktiga på det ännu, men vi har i alla fall en embryo här. Det finns ingen riktig dokumentation än om jag minns rätt. Jo, lite, men inget som man direkt kan stoppa i händerna på kunderna, utan det är mer för att kunna svara på kundernas frågor, men det finns ingenting som hindrar att du skulle kunna skriva dokumentationen här. Fast idag gör vi inte riktigt det. Kanske bara *release notes* som skulle kunna gå till kunden.

För att i Confluence har du möjlighet att exportera ut det till antingen PDF- eller Word-dokument, för det är så vi brukar göra, att exportera det.

Det verkar smidigt.

Det är jättebra. Genom att bara klicka får jag en PDF direkt nu där vi ser samma sak som vi editerat precis i Confluence nu och det blir snyggt, tycker jag.

Så det är rätt så mycket av arbetet man kan göra med hjälp av wiki?

Absolut. Vi gör i stort sett all dokumentation förutom användarmanualer. Så vi tror starkt på det. Det finns nackdelar också. Just Confluence har sina nackdelar, men fördelarna överväger nackdelarna.

Vilka nackdelar är där då?

Det som retar folk mest här är att man inte kan söka med en inledande stjärna (*). Man kan söka med en bokstav och sedan en stjärna, men man kan inte ha en inledande stjärna. Det är en ganska liten sak, men det retar upp folk.

Den är inte hundraprocentig heller när man vill editera en sida [visar på skärmen]. Här är två lägen. Detta är Word-liknande och sedan finns det det klassiska *wiki-markup*-läget. Wiki-markup-läget är exaktare. Då behöver man inte omvandla utan det blir vad det blir. Dock har detta läge högre inlärningströskel. I Word-läget är det man retar sig på att det inte alltid blir som man tror. Ibland har den svårt att omvandla den här *rich texten* till wikitext.

Där finns även vissa problem att importera Word-dokument. Det funkar sådär halvbra. Det ser snyggt ut, men det är nedlusat med referenser. Det ser ut som om Word försöker göra en HTML-sida. Det blir inget vettigt av det.

Trots den höga inlärningströskeln, går det ändå att övertyga folk om att använda wikin?

Ja. Så länge rich text-läget finns så går det att övertyga folk om att använda den. Sedan beror det lite på vem som gör det också. Har du själv kodat så märker du problem med rich text-läget. Håller det högt upp, så att säga en säljare till exempel, hade han/hon aldrig gått in i wiki-markup-läget, men det är bra att båda lägena finns.

Är det hela företaget som använder wikin?

Ja, i stort sett. Jag vet inte om ledningsgruppen använder den, men de söker definitivt information i den. Supporten använder den jättemycket, utvecklingsavdelningen och konsultavdelningen använder den rätt mycket. Marknadsavdelningen vågar jag inte säga mycket om. Sälj och marknad använder den inte mycket för att dokumentera, men de söker säkert mycket information i den [tittar i wikin]. Nej, jag ser ingenting som rör försäljning. De använder nog fortfarande mycket Sharepoint. Vi har även kunder i vår wiki. Vi har vissa spaces, som vi och kunder delar på, där kunderna får komma in och dokumentera och även läsa dokumentation.

Så det finns möjlighet att ställa in olika rättigheter på olika spaces?

Ja, absolut, både för grupper och individer.

Hur är det med supportärenden?

Ja, de använder wiki och Jira för att hantera ärenden.

På tal om Jira så är ju Confluence och Jira är utvecklade av samma företag; Atlassian, som du nämnde innan, så vi kan tänka oss att de är bra integrerade.

Ja, det funkar jättebra.

Vilka andra verktyg, förutom Confluence och Jira, används inom systemutvecklingen?

Vi har vissa andra från just Atlassian [visar på skärmen]. Just det, bara för att visa integrationen. Den här bilden kommer från Confluence och vi är inne på en *dashboard* på Jira. Där kan jag hämta information om ett projekt och presentera det och vice versa. Här borta kan jag då lätt visa information ifrån Jira som man ser här nere. Här är den release som kommer. Här direkt då, har Jira listat *issues*, alltså vad som kommer släppas i nästa release. Här kan jag gå in i Jira och titta på just den *tasken*. Den integrationen är jättebra.

Sedan använder vi även *Fisheye* från Atlassian för att svara på din fråga. Fisheye används för kodhantering, Java-kodshantering och *review* utav kod.

I vår versionshantering använder vi CVS (Concurrent Versions System) och till viss del Subversion än så länge. Det är då kopplat direkt in till Fisheye. Om vi går in här, så ser vi att en medarbetare i Polen har checkat in nånting till GCM (Global Customer Master) här. Man kan hålla koll på utvecklingen av koden och jag kan gå in och se vad han har gjort och man kan se skillnaden till hur det såg ut tidigare. Den här håller all koll på all *incheckning* som sker i CVS. Man kan se åt andra håll också, säg, jag kanske har en *Jira-issue*. När jag får den, kan jag se hur specifikationen för den är gjord i Confluence, så det är bra ihoplänkat. Sedan har vi en Atlassian-produkt till, tror jag...

Vi hade en innan; Cruise Control, som byggde koden på natten så att man på morgonen ser om det har funkat, men den har vi bytt ut för att vi tyckte inte att den var optimal. Nu har vi *Hudson* i stället som gör samma sak, nämligen att den bygger all kod som ligger i CVS och så presenterar den hela rapporten på morgonen.

Hudson kör även tester som vi definierat upp som ska köras på koden.

Hudson?

Hudson, ja... [visar på skärmen igen]. Här ser man GCM-projektet som jag jobbar med. Här byggde vi senast för 6 timmar och en minut sedan och här kan man se resultatet på hur det gick.

Finns det någon sorts kunskap i era projekt som är svår att överföra eller få in på wikin, som du känner till?

Ja. Går det bra med text är wiki smidigt. men det är just som alltid: Är det någon som slutar, om man inte skrivit ner det, så finns det inte en chans att någon annan får tag på den. Även nerskriven, så är det inte heller säkert att man får tag på den eller att man förstår vad personen skrev ner. Det är just mjuka kunskaper som är svåra att få ner. Vi har därför vissa standarder hur man skriver kod, hur man skall bygga upp SQL-satser och liknande eller hur man bygger saker om man är hos kunden.

Vissa saker är då svåra, men de är svåra för att de är svåra att få ner i text. *Det har inte med just wikin att göra, utan det hade varit lika svårt att skriva ner det i Word*. Möjligtvis är wikin till och med lättare att ändra i och lätt att *accessa*. Jag kan inte komma på ett verktyg som skulle vara bättre heller.

Ibland är det bra att kunna rita saker och ting och det är faktiskt svårare på wikin eller om man inte har något *Visio*-liknande verktyg på wiki direkt. Det finns vissa *plug-ins* till Confluence, bland annat *Gliffy* som vi har pratat om innan, men vi har inte använt dem så mycket. Just att rita på en dator kan vara svårt om man inte är duktig på det, tycker jag. Är du grafiker, så är du bra på att rita på datorn. Gör du det bara ibland, så är det lättare att göra det på en whiteboard, för du är van att använda en penna. Det är svårt, absolut.

Hoppas det blir bättre verktyg för att rita. Gliffy är ett intressant plug-in.

Ja, det är som Visio, eller PowerPoint. Det är nästan samma sak.

Tekniken utvecklas, bland annat HTML5 som ger mer möjligheter att skapa sådana typer av applikationer och tillägg.

Ja, precis, det kommer säkert. Confluence släpper rätt ofta nya versioner. Kanske lägger de till denna funktionalitet. Det är dock rätt dyrt med Confluence. Smart prissättning egentligen för att ta halva sin initiella kostnad per år i *maintenance*. För det kostar 4000 dollar per 500 användare. Då tar de 2000 dollar per år i maintenance för att få deras system uppe. Det är rätt saftigt, så de är tvungna att släppa nytt så att folk vill betala.

Men det ser väldigt smidigt ut.

Ja, det är snabbt och enkelt att använda. Det är jättebra. De lägger till nya funktioner och integrationen blir bättre och bättre.

Du sa innan, att du ser det som din vision att föra fram wikin så mycket som möjligt.

Ja, jag är väldigt *pro* Confluence.

Utan att nämna några namn, vem är det mer på företaget som driver igenom detta?

Ja, det är utvecklingschefen, han driver också på. Han tycker det är jättebra. Att det ska dokumenteras på wikin, är drivet av ledningen, för att det är viktigt för supportärendena. Om man dokumenterar noga, kan man ha nytta av det i framtiden. Om det är någon som är duktig och bara fixar och ingen vet vad som har gjorts, så spelar det ingen roll vilken wiki det är då.

För mig är det viktigt att vi lämnar Sharepoint så fort som möjligt, framförallt den gamla versionen som vi använder. Så är det dumt att ha två. Det är värdelöst, för då vet man inte var informationen finns. Det är nog det viktigaste. Förra jobbet, där jag jobbade ett halvår, där hade vi fyra stycken olika wikiverktyg, tror jag, dels Confluence, dels Sharepoint, och sen något annat som jag inte ens vet vad det heter. Så det viktigaste är att man har det på *ett* ställe, än vilken wiki det är för att folk ska slippa leta runt och de hittar inte det.

I vilken grad påverkar Syncrons spridning (att det finns kontor i Warszawa och Tokyo) användandet av wiki?

Jag tror att det är jätteviktigt. Vi har en kille som har precis börjat hos oss, en utvecklare, en duktig utvecklare sedan 10 år tillbaka. Han säger att han är helt imponerad att vi har fått in så mycket information på vår wiki som det finns. Han hittar så mycket och det är så att utvecklingsteamet nere i Polen tar till sig wikin och vill verkligen dokumentera. Det ligger i deras intresse också för att slippa ställa frågor hela tiden. Måste man fråga varje gång man gör någonting, då blir man störd i det man sysslar med just då. Dessutom är det så att det är många länder, gällande Confluence, att du då skriver du inte på polska eller på svenska. Du skriver på engelska. Det är regel för allting. Där är det självklart.

Ibland kan du hitta en PowerPoint på svenska, men det händer inte på wiki, eftersom du vet att den läses av japaner, amerikanare, tyskar, svenskar och polacker.

Vi har inte heller någon *fileshare* som vi har dokumenten på. Det hade vi tidigare innan vi införde det här. Alla vet att det finns en *inmappad drive* som heter S:\ där dokumenten ligger, men det är bara en fileshare och du kan inte söka särskilt smart i den. Hur smart Windows är på att söka, det vet vi ju, men när du söker här så söker du överallt. Du hittar precis som du *googlar* på nätet. Så det är jätteviktigt för oss eftersom vi är så spridda nu. Det hade funkat med en fileshare, om det är ett bolag i en stad som det var från början och du kan alltid gå och fråga, men jag menar, vi sitter ju i olika tidszoner, olika kontinenter, med olika språk och med olika folk som gör olika saker. Det hade inte funkat vettigt för oss, om vi inte hade haft något sådant här. Vi är väldigt beroende av att det fungerar. Någon gång var det någon som hade för mycket rättigheter att lägga in ett plug-in själv. Det var något plug-in som inte funkade riktigt, vilket sänkte hela Confluence. Då hörde man väldigt fort att Confluence verkligen användes. Folk ringde från från olika avdelningar och kunder ringde och talade om att det var nere. Ni märker väl, att jag är väldigt positiv till en wiki.

Ja, vi märker det. Det du har berättat och visat oss, visar på att wikin verkligen används.

Ja, det finns hur många sidor som helst, som bara beskriver små saker, som folk som lägger ner tid på att beskriva hur saker funkar. Här [visar på skärmen] är en beskrivning utav *Java-bönor* och hur det funkar i GUI:t. Det är lätt att dokumentera. Visst, du kan ta en skärmdump och lägga till pratbubblor med kommentarer. Det är också smidigt. Det är lätt att lägga till bilder. Då går du in på menyn och *attachments* och bara laddar upp en bild som är lätt referera till på rätt sida. Det är väldigt smidigt.

Används wikin inom alla projekt, alltså?

Precis. Vissa projekt använder wikin lite sämre, däribland vissa kundprojekt, men fördelarna märker kunderna rätt så fort. Det sprids som ringar på vatten. Jag har varit med om ett projekt som skulle ta fram en ny utvecklingsmetodik för vår konsultorganisation. Där har vi definierat att vi jobbar med Confluence, att alla *case* ska upp och beskrivas. Man ska inte göra någonting som inte finns dokumenterat på Confluence. Då vet du allt om projekten och du kan spåra tillbaka ifall något går snett. *Just spårbarheten och att slippa gissa och leta är de främsta fördelarna*.

Har valet av systemutvecklingsmetod påverkat sättet hur ni dokumenterar? Det finns till exempel vissa metoder där man har sagt att man ska strunta i dokumentation.

Hmm, scrum brukar beskyllas för det, men jag tycker det stämmer inte riktigt om man ser hur man jobbar i scrum. Scrum försöker ha mindre dokumentation för att utvecklingen ska ledas av utvecklare. För att svara på din fråga, så vet jag inte riktigt. Vi har haft projekt som vi har drivit mer eller mindre i vattenfall plus att vi använt scrum i vissa, mellanting i andra och alla inser vikten av att ha en bra dokumentation. Sen å andra sidan stöder inte Confluence riktigt scrum. Folk sitter i Polen, här i Malmö och i Lund. Det är omöjligt att ha en fysisk scrumboard för projektet. Visst, i Polen där det utvecklas, har de en fysisk scrumboard. Lite bättre systemstöd för det hade det varit bra. Jag märker det börjar komma mer och mer sådant, bland annat plug-ins till Confluence. Det finns till exempel ett helt program från Atlassian som har stöd för att jobba med scrum och andra agila utvecklingsmetoder. Det är nog där man behöver mest stöd just nu. Det finns en hel del bolag de köper. Det går hur bra som helst för dem. De tar ju också så mycket betalt. Här då [visar Confluences hemsida] finns det *Greenhopper*, ett verktyg för agila projekt. *Bitbucket* funderar vi på att använda för versionshantering istället för CVS, vår egen versionshanteringsserver.

[...]

Har vi förstått dig rätt att just det faktum att Syncron är spridd över länder och kontinenter driver användandet av wiki? Även om det finns på scrumboard måste det skrivas in på wiki för att alla involverade ska ta del av informationen.

Exakt. Om alla inte är på samma plats är det svårt att köra fysisk scrumboard och även spridning av information är ofta ett problem. Har du det bara på en fysisk scrumboard, är det ett problem. Det är lätt på sitt sätt också. Hos en av våra kunder så jobbar man med ett Excel-ark. Lite halvsmidigt, men det funkar ju. Du kan checka ut vissa uppgifter du ska jobba med, men just att alla ska få tillgång till informationen, eftersom vi inte är ett jätteföretag, men vi har folk överallt, måste vi blanda från olika ställen på projekten. Eftersom det mesta av utvecklingen sker i Polen och jag är Business Analyst i projekten vill jag veta hur det går där nere. Utvecklingschefen sitter här i Malmö. Vårt distribuerade företag måste ha bra sätt.

Även om det kommer en ny på företaget, måste det underlätta för den att komma in?

Ja, precis, som den nya killen här. Han jobbar inte med utveckling, men han har arbetat mycket som utvecklare tidigare. Han kan gå in på wikin och se hur vi jobbar med CVS, med *Eclipse* och hur man bygger produkten. Innan så fick man prata med någon annan. Nu ligger allting här. Bara läsa några instruktioner så är du på banan direkt. Det gillar jag.

Det är mycket intressant. När vi berättade för våra kompisar och bekanta om vår uppsats och om just ämnet wiki så reagerade vissa med att det inte var aktuellt längre, att det var "old news". Hur känner du inför detta?

Ja, företag är lite sena. Det är precis som med Skype. Det har funnits länge. Det är bara de senaste åren som företag har börjat använda detta. Nu har vi insett hur bra det är. Det är likadant med wikin. De har inte varit mogna för att använda den. Vi använder wiki nu.

Vi vill ha mer och mer funktionalitet som vi får i Confluence. Det finns wikiverktyg på nätet såsom Wikipedia, men det är en stor skillnad mot en sådan wiki och vad vi har på Confluence.

Det verkar som att Confluence innebär mer kommunikation och ett bättre samarbete som sänker administrationskostnaderna.

Har ni kollat på Asana? Killen bakom det är en av Facebook-grundarna. Googla på det. Det är väldigt intressant vad det gäller informationsspridning. Där har en *feed* från alla nyheter från projekten där du är med. Confluence har det också, fast det är inte riktigt där än. Här har vi feed på allt vad folk gör, på varenda projektdel som finns i Confluence. Sedan kan du lägga in dina projekt lokalt, men det de har gjort är att de har byggt ihop det. Så att du har mail, du har *tasks* som kommer och även scrumboard är integrerat på ett väldigt snyggt sätt. Informationsspridningen är det stora problemet. Informationen finns någonstans, men du hittar det inte.

Det låter lite som att para ihop Confluence och Facebook.

Ja, precis och även få med Jira och din mail i samma program. Lite åt det hållet.

Ska inte Confluence bygga in den här funktionaliteten?

De kommer kanske dit, men de är rätt långsamma.

Det hade inte varit dumt om man kunde gå in och "gilla" de projekt som man vill få nyheter ifrån.

Ja, exakt, du kan lägga till något som din favorit. Då får du bara en kortare lista. Här finns det att du kan gå in och sätta prioriteringar på *issues*. Så det finns lite små embryon. Googla i alla fall Asana. De har en snygg demo på nätet. De har inte släppt något officiellt än, men det finns en bra demo.

Är det några andra företag som ni ska intervjua?

Vi har intervjuat ett litet företag i Lund som inte använder wiki så mycket internt. Det har kanske med projektstorleken och dess spridning att göra.

Förmodligen. För ju mindre du är, desto mer tid måste du lägga på just att verkligen generera pengar och då är dokumentation en kostnad i alla fall initititalt. Vi har ändå kommit i den storleken att, dels vet vi att vi kan spara pengar på att dokumentera, och dels har vi den overheaden som vi kan jobba med så att vi kan dokumentera med wiki, så att vi tjänar pengar ändå.

Kom gärna mer fler frågor. Jag brinner för det ämnet. Jag tycker det är roligt.

Det tycker vi med. Har du något annat som du vill ta upp?

Nja, en sak som jag tycker är viktigt med wiki är att just det faktum att man har *egen kontroll*, att man inte är hänvisad till kundens dokumentationssystem.

Så sant som det var sagt. Tack så mycket för intervjun.

Tack själva.

Intervju med Hampus på Bool

Den 18 april 2011

Beskriv lite kortfattat vem du är, vad du jobbar med och hur länge du har jobbat med detta.

Hampus heter jag. Jag är en av grundarna och delägarna i Bool som vi startade år 2008, på våren där. Jag hade jobbat ett år tidigare med systemutveckling, innan Bool blev Bool. Egentligen samma företag, men lite andra ägarstrukturer och då inriktat på .NET-utveckling.

Vad hette det?

SA-Group hette det. Det blev Bool 2008, med lite nya ägare och ett helt nytt koncept med ny inriktning och alltihopa. Så det är tre år nu vi har varit verksamma och jag har bakgrund precis som ni som läst systemvetenskap, där jag tog magister i det. Jag själv är inte superintresserad av teknik. Jag är nog den enda på Bool som inte är det, förutom Peter möjligtvis, vår säljare. Eller jo, jag är väldigt intresserad av teknik, men jag är inte så praktiskt lagd. Jag kodar liksom inte. Det är inte min grej, men jag tycker det är väldigt kul med teknik och kunna applicera den.

Jag har haft lite varierande roller här på företaget genom tiderna. Jag har jobbat väldigt mycket med administrativa uppgifter såsom att ha tagit hand om ekonomi, lön och egentligen alla delar som inte är kodning. I ett litet bolag måste alla som kan producera och debitera. Så jag har tagit hand om alla möjliga uppgifter.

Det är ganska många uppgifter kan vi tänka oss.

Jo och jag har varit projektledare i en del av projekten också. Just har vi växt lite och vi börjar mer och mer jobba mer med vår profil, hur vi presenterar oss, alltså genom olika marknadsaktiviteter. Mina roller är att dels sköta ekonomi och ekonomistyrningen och dels hantera marknadsföringen. Så jag är ekonomi- och marknadschef.

Du har redan kommit in lite grann på bolaget? Vad utvecklar ni på Bool?

Som ni säkert redan känner till så jobbar Bool med Sharepoint och portaler och enbart detta.

Hur sköter ni er systemutveckling? Finns det några speciella metoder som ni använder?

Ja, vi är rätt unga allihopa. Så vi har kommit med nya skolan med agila metoder och vi har ett agilt tänk. Från början använde vi scrum rätt mycket, så som scrum var tänkt att användas, men efter ett tag, när man har gjort en del projekt, började vi fundera att det var kanske inte optimalt för just vårt företag att använda scrum rakt av. scrum är egentligen till för *teams* på 4-5 personer. Det ska också gärna pågå i flera månader i längre projekt. Det kräver liksom en struktur som vi inte riktigt har på vårt företag.

Så vi kikade lite grann på hur vi skulle kunna anpassa det här eller ta fram vår egen metod som är anpassad för våra egna projekt, alltså Sharepoint-projekten, och för vår storlek. Så vi tog fram eller rättare sagt, Johan, vår tekniske ansvarige tog fram en egen metod som är helt anpassad för vår verksamhet, som heter *Boolean*. Den bygger på scrum, kanban och lean-tänket. Så att i stället för att ha ett scrumboard med arbetsflöde och backlog och processerna som man går igenom till leverans och det är iterativt då med cykeln...

Vårt företag då, dels sitter många ute som konsulter, alltså är de inte här. Det finns många små projekt, där det är bara en man som sitter med det. Vem sitter då där och har stand-ups med det teamet? [skrattar]. Det går liksom inte och sedan är det så att det kommer väldigt många supportärenden och sådant här...

Alltså ingen fast backlog, som man måste avvakta, utan ett löpande flöde som kommer in och som aldrig tar slut.

Det ska hanteras och sedan är det alla säljstödsaktiviteter som vi också måste resursbelägga och hantera med våra resurser och styra upp. Det kan vara det här med att ta fram offerter, men det kan även vara demos, säljmöten och presentationer. Allt som konsulter är involverade i och detta ska inbegripas av en och samma metod. Så därför tog vi fram Boolean, för att dels hantera större projekt. Den är ganska lik scrum som också hanterar det här flödet med löpande aktiviteter och supportärenden och säljaktiviteter.

Den har vi kört rätt länge och den fungerar rätt bra och det bygger mycket på att det visualiserar flöde: vi sätter upp lappar, bryter ner det och gör det synligt. Så det bygger mycket på de principer som finns i agila metoder.

Angående kunskapshantering, hur dokumenterar ni? Hur får ni ner kunskap just i utvecklingsprojekt?

Vi har gått ifrån ett pärmsystem som vi hade i början. Då skrev vi ut saker och dokumenterade det i pärmar och la in alla lösenord som behövdes på olika servrar. Det har vi förfinat efterhand. Numera har vi en egen *projektsajt* för varje projekt på nätet. Där lagrar vi alla dokument, all korrespondens från kunden och alla kravspecar från kunden. Allt som produceras i dokumentform hamnar där. Där ligger också forum och lite uppgiftshantering. Så vi har det som projektplats där vi samlar alltihopa. Det är då versionshanterat och backupat. Vi har märkt att det blir väldigt problematiskt när man gör ett projekt, sen går någon ur projektet, sen kommer in någon annan som ska ta supportärenden eller om man ska gå in och göra någonting. Den tiden som det tar att sätta sig in för den personen, om man inte har tillgång till allting, är väldigt lång. Det är väldigt viktigt att ha en plats där man har allt dokumenterat.

Är det oftast detaljerad eller generell dokumentation?

Det är både och. Det kan vara väldigt övergripande dokument som har producerats i början av ett projekt av kunden. Det är sådana väldigt grova beskrivningar, alltså målet med projektet som kunden vill uppnå. Efterhand byggs det upp, så det kommer mer detaljerade kravspecar som man lägger ut där.

Det kan vara systemdokumentation som kan ligga där också. Alltså, det är ingen klassificering av dokumentation alls, utan vi lägger ut all dokumentation som finns på en och samma plats och det läggs ut kontinuerligt.

Använder ni någon sorts wiki på Bool?

Ja, det gör vi. Vi har det, inte för varje projekt, men däremot har vi en generell wiki, som är en teknikwiki. Där skriver alla allt, alla nya områden som man stöter på. Så tanken är att så fort någon har utforskat okänd mark inom vårt område så ska man skriva ner de erfarenheter man hittat och det är bara en *brain dump*. Det ska inte vara strukturerat. Därför är det en wiki. Det ska inte vara för formaterat. Det ska gå snabbt. Där ska man helst skriva ner alla erfarenheter man får så att alla andra kan ta del av det. I den wikin så är det en intern infrastruktur med våra egna servrar och lösenord för att kunna komma åt den. Genom att läsa där, ska man kunna sköta sitt arbete på ett bra och effektivt sätt. Där ska finnas så mycket information som möjligt.

Har ni haft wikin sedan ni startade?

Jag tror den har varit igång i ett år, kanske.

Vilken programvara använder ni för wikin, alltså vilken sorts wiki är det?

Det är Sharepoints egen wiki.

Finns det någon kunskap som är svår att få ner eller få in på wikin?

Ja, allt som inte är tekniskt egentligen. Alla de här mjuka delarna; kundkontakt och kunddialog. Man kan skriva en mötesprotokoll, att man har kommit fram till det och det, men det här att man snackar med kunden, att man tar en kopp kaffe med kunden, alltså de mjuka delarna är väldigt svåra att få ner på papper eller i text. Det är väl det.

Är det några speciella projekt wikin används i?

Den används i alla projekt egentligen för den är inte knuten till något speciellt projekt alls. Man skriver inte ner dokumentation som är projektspecifik. Det är mer så att, jobbar med något eller löser man ett problem i ett projekt, då tar man det mer generellt och lägger in det i wikin, alltså mer konceptuellt, vad det är för typ av problem och hur man löst det och vad man ska tänka på.

Vilka andra verktyg använder ni förutom wikin för att samarbeta?

Vi har Microsoft Office Communicator, som egentligen är någon slags MSN-chatt på arbetsplatsen. Den använder vi väldigt flitigt. Det är inte så mycket för att lagra kunskap utan det är för att konsulterna ska kunna samarbeta och kommunicera med varandra i realtid. Det blir det att man kanske ofta sitter i olika projekt. Man kan inte alls sitta och ringa varandra hela tiden. Det är väldigt smidigt att ha en klient som man ser alla online så att man kan slänga iväg en fråga.

Finns det någon koppling mellan Communicatorn och wikin?

Nej, egentligen inte. Det är dock integrerat på så sätt att varje gång man ser ett namn, ser man en symbol bredvid om den är online så att man kan ta kontakt med personen. Så om någon har författat någonting, så kan jag se en symbol där, säg, Pontus har skrivit någon grej i wikin, som jag kanske inte helt förstår, så kan jag väldigt enkelt, genom att klicka på ditt namn, starta en ny chatt, skicka ett mail, eller någonting.

Hur är det att utveckla med hjälp av wikin jämfört med hur det var innan?

Just wikin ersätter inte gamla pärmar, men att ha *projektsajter* där allting finns online är väldigt smidigt. Det är väldigt lätt att ändra och skriva. Det är väldigt lätt administrerat. Det ligger centralt så att alla kommer åt och allting finns där. *Just wikin har ju kommit till när vi har vuxit och expanderat, så den har börjat fylla ett behov ju fler vi har blivit.* Det kan inte vara något jättestort behov av att ha det om det är ett par stycken som utvecklar, men ju mer spridda projekten blir, ju fler personer som kommer in i bolaget och ligger på olika erfarenhetsnivåer, så har vissa inte jobbat med det länge medan vissa har jobbat med det rätt länge. Så att det är ett verktyg att skriva ner kunskapen och erfarenheterna så att fler kan ta del av det.

Finns där några brister med wikin som du kan komma på?

Även om wikin är ett jättebra och snabbt sätt att skriva ner saker så tar det ändå en del kraft. Om man sitter mitt uppe med någonting, så måste man ta 5 minuter av sin extremt värdefulla tid och lägga den på att skriva ner något som andra kan ta del av. Det är kanske det bästa sättet som finns, men jag vet inte. Det tar ju ändå tid. Det är samma sak med vanlig systemdokumentation som man kanske gör helt i slutet när det blir nödvändigt, när kunden ber om det. Det är kanske inte något man gör med lapparna alltid.

Tror du valet av systemutvecklingsmetod, Boolean i ert fall, har påverkat sättet att dokumentera?

När det gäller dokumentation till kund så påverkar det nog inte så mycket. Jag tror det sker på samma sätt. När det gäller dokumentation för sitt eget bruk, när man skriver ner sina egna erfarenheter så blir det ju annorlunda. Just att man jobbar agilt gör att man levererar saker, sedan påbörjar med nya saker. Det är mycket delprojekt så man bryter ner det. Just när man avslutar en sak så har man ett tillfälle att titta tillbaka: vad gjorde vi de här två-tre veckorna? Det är då man reflekterar över det. Man jobbar mycket med utvärderingen: hur arbetade vi de här två veckorna? Hur gick det? och sedan skriver man ner det. Så det hänger ihop lite grann alltså metod och dokumentation.

Boolean är ju bland annat kanban. Uppgifterna som man skriver upp på post-it-lapparna, finns de också på projektsajterna?

Nej, de hamnar bara på lappar egentligen. Vi sparar alla lappar som vi bryter ner, men de sparas inte på den nivån. Det är mer utvecklingsteamet som har ansvar för lapparna.

Hur länge sparas de?

Tja, jag tror inte vi har slängt några någon gång, men man brukar aldrig titta på dem igen ändå.

Tror du man kommer behöva titta på dem någon gång?

Jag tror om man ska göra något projekt där man har liknande moduler som ska byggas, kan man ju väl gå tillbaka och titta hur man gjorde tidigare.

Det finns kanban-tavlor som man kan använda digitalt. Är det något ni på Bool har funderat på?

Jag vet inte. Det är egentligen mer Johans avdelning, det där. Jag tror inte det just för att det ska ju vara ett verktyg som är så extremt tydligt, som ska sitta väldigt konkret. Att du måste flytta en lapp, det måste nog följas mycket bättre i live-bruket. Jag tror att sitta och göra det digitalt gör att det inte blir riktigt samma effekt av det.

Men om Bool utökar och det blir fler anställda som sitter spridda, blir det inte svårare det att ha en fysisk tavla?

Jo, det blir det, absolut. Boolean var ju framtagen i ett visst skede där vi var ett visst antal personer och den fungerade optimalt som det är nu eller som det var då. Boolean är en av de *nyckelframgångsfaktorerna* på Bool. Regel ett i Boolean är att den ska utvecklas kontinuerligt. En del av metoden är att utveckla sig själv. Så att alla som använder metoden ska utvärdera metoden kontinuerligt och anpassa den efter hur situationen ser ut. Så Boolean är ingen bok som man har skrivit och följer den utan det skrivs ett nytt kapitel varje dag och den utvecklas hela tiden.

Finns den nedskriven?

Ja, det finns den. Det finns ett dokument kring den, som förklarar principerna och hur man jobbar med den, men det har hänt ganska mycket med den sedan vi lanserade metoden fram tills idag. Olika *lanes*, som vi kallar dem, har bytts ut. Vissa processer har bytts ut mot andra. Alltså det utvecklas. Så är det ju.

Dokumentation för kunderna, hur tar ni fram dem?

Det blir ju så att man försöker varje gång det är ett nytt projekt att dokumentera varje steg. Det gör man till viss del, men det blir alltid mycket jobb i slutet för att dokumentera ner det. Till exempel installationsmanualer och sådana här bitar. Sådant är väldigt svårt att skriva innan. Det är mycket så att man sätter sig och skriver ner mycket när man närmar sig slutet av projektet, men varje modul, eller delmodul, brukar dokumenteras löpande. Fast själva det som man lämnar över till kunden skrivs kanske till 70 procent i slutet.

Vad finns det för andra system, förutom Sharepoints inbyggda verktyg för att hantera supportärendena, som ni använder?

Vi har en supportkedja, kan man säga. Vi har en supportmail som vi använder för att hantera supportärenden. Det går ut på att den som är ansvarig i projektet, vid den tidpunkten, tar emot ett supportärende och *dispatchar* ut det till utvecklarna och det har ofta varit Johan när han är på plats för han är ansvarig. Han får in ett supportärende. Han gör en lapp på det, sätter upp den på tavlan och sedan kollar han hur resurshanteringen ser ut på tavlan, hittar en lämplig resurs och tilldelar detta till den resursen alltså sätter han en ikon, en namnskylt. Sen så att man lägger in en ikon och prioriterar den. Vi har också olika färger på lappar. Röda lappar ska egentligen aldrig användas om det inte är akut och håller på att brinna. Är det verkligen något akut, kan man göra en lapp som går före i kön.

Man kan tänka sig att om man skulle använda röda lappar för ofta skulle hela systemet sluta fungera.

Ja, precis.

Så när ett supportmail kommer in skrivs en post-it-lapp och sätts upp på tavlan. Kommer det någon motsvarighet på projektsidan då samtidigt?

Det beror lite grann på projektet då. Vissa projekt blir större och får sitt eget liv. Då kan det finnas egen ärendehantering för specifika projekt, så det beror på projektet. Boolean är egentligen det sättet man tar hand om supportärendena.

När vi pratade om wikin sa du att det bara var brain dump. Så det finns inga begränsningar hur man skriver?

Nej.

Har du upplevt att det saknas struktur och att det är svårt att hitta information då?

Jag har inte jobbat så mycket med teknik, så jag har inte varit där så mycket och kollat. Jag tror det kräver ett visst ansvar av de som jobbar med den att skriva på ett strukturerat sätt så att man kan hitta, men vi har inget regelverk för hur man ska använda wikin utan vi tycker att det är bra om så många som möjligt skriver där. Fast vi är inte så stora. Hade vi varit många fler som hade jobbat med det skulle det nog behövas lite mer regler på hur man använder den så att det inte blir kaos, men wikin ska ju vara lite vildvuxen, liksom. Det är mer till att det ska vara lite *kantigt*.

Används wikin, eller intranät överhuvudtaget, för ekonomi och marknad?

Ekonomi... inte så jättemycket. Det sköter vi på ett annat sätt, men för marknadsaktiviteter definitivt. Alla presentationer, alla konsult-CV:n, alla typer av aktiviteter och afterwork-listor. Allt med det lägger vi ut på intranätet och gör det tillgängligt.

Det är du som organiserar marknadsaktiviteter, eller hur? Hur gör du då? Säg att du har skrivit en gästlista. Lägger du den på intranätet direkt?

Ja, det gör jag och speciellt PowerPoint-presentationer. Det kan finnas ett antal formella bolagspresentationer, men egentligen varje möte, varje aktivitet kräver någon anpassad presentation för att passa in i situationen. Så man bygger upp ett *bibliotek* med massa olika presentationer och varje med sin egen *tweak*, liksom. Så har man en hel lista med gamla presentationer kan man till slut bara plocka från varandra och det går snabbt att skapa något nytt.

Säg man vill hitta en specifik presentation. Är det då lätt att söka?

Ja, det är absolut lätt att söka. Det är väldigt mycket metadata på det, så att man söka på vem som har gjort det, vilken tidpunkt, när den är gjord, per kund eller per situation, men det är inte så många, så att man kan bläddra ganska snabbt

Har Bool någon gemensam hårddisk också?

Jo, det har vi. Vi har en så kallad *Common Storage* där vi lagrar större filer. Det är massvis med virtuella maskiner. Kanske någon kund har något projekt som ligger på en egen virtuell maskin. De är väldigt stora, så vi lägger dem på ett ställe. Där lägger vi även massa installationsfiler och programvara. Det kan vara Office-paketet eller sådant. Alltså det är mycket virtuella maskiner och programfiler.

Men inga presentationer?

Nej. De ligger på intranätet.

I vår uppsats har vi funderat på vad en wiki är. Enligt definitionen ska en wiki vara ett "webbaserat verktyg där alla involverade kan redigera och se resultatet direkt". Sharepoint kan också passa in under en del av definitionen, eftersom där kan man alltid klicka på "redigera sida" och alla ser ändringar direkt.

Ja, det är sant.

Tack så mycket för intervjun. Det var snällt att du ville ställa upp.

Tack själva. Det var intressant.

Intervju med Mikael på Haldex

Den 19 april 2011

Beskriv lite kortfattat vem du är, vad du jobbar med och ungefär hur länge du har jobbat med detta.

Jag heter Mikael Hansson och jag är ansvarig för den gruppen som heter *System* eller *Systems* inom Haldex. Vi är en funktion som har två grupper. Den ena gruppen heter ERP och det är där vi har hand om vårt affärssystem, EDI (Electronic Data Interchange, överföring av strukturerad information enligt ett överenskommet format) och vår kommunikation. Den andra gruppen heter *Applications* och där har vi Thorbjörn här [pekar på sin medarbetare] som är chef över den gruppen. I den gruppen ingår alla övriga system, alltså allt som inte är affärssystem. Det kan vara alltifrån affärsapplikationer till operativsystem... ja, alla program som körs på datorer egentligen.

Jag har varit på Haldex i tre år nu. Först jobbade jag som konsult och var ansvarig för ERP inom Europa. Jag ansvarade för Axapta, numera Microsoft Dynamics AX. Jag var då ansvarig för att skapa *releaser*, göra ändringar och ha hand om supporten. Det var stort och smått. Alltifrån att konfigurera systemet, med parametrar och sådant, till att ta fram ändringsbegäran, men också till att lägga upp nya användare.

Som sagt, så arbetade jag som konsult innan, men jag har också varit fast anställd. Jag har egentligen haft nästan alla positioner [skrattar] inom IT-organisationen alltså. Jag började som utvecklare i en minidatormiljö och i PC-miljö. Ganska länge höll jag på med det och jag gjorde då mest affärsapplikationer. Jag har även jobbat med att ta fram rapporter och jag har varit metodstöd och projektledare och så småningom så arbetade jag som IT-chef.

Jag har hållit på med det här sedan... tja, min fru säger att jag inte får säga detta, men egentligen sedan datorerna kom [skrattar]. Jag var med när PC:n kom och jag hade jobbat lite med de här grejerna redan innan.

Så jag har hållit på med det mesta.

Berätta lite kortfattat om företaget och er produkt/tjänst.

Vi är sedan någon månad tillbaka en mer global IS/IT-organisation eller ICT (Information and Communications Technology) som vi nu kallar oss för. Så nu har vi ett mer globalt ansvar, som är ännu mer tydligt än vad det var innan. Jag och Thorbjörn här är ansvariga för alla systemen inom Haldex.

Haldex är ett tillverkande företag inom fordonsindustrin och vi finns på en massa olika platser, globalt då. Landskrona är vår *sajt* i Sverige och sedan har vi även i Frankrike, Tyskland och Ungern i Europa, men vi finns även i USA, Mexiko, Brasilien, Kina, Indien och sedan finns det en massa försäljningskontor i en hel del andra länder också.

Vi gör tillbehör till bromsar, kan man säga, och *broms adjusters* och länkarmar och sådana saker. Jag nöjer mig med att säga att vi jobbar med CVS (Commercial Vehicle Systems) och med bromsar. Ni kanske vet att vi håller på att dela upp företaget.

Vi har sålt just fyrhjulsdriften till ett amerikanskt företag och det som hette Hydraulics tidigare delar vi också ut nu till aktieägarna. Kvar är alltså den delen som heter Haldex Brake.

Hur sköter ni er systemutveckling?

Ja, jag tycker inte att vi har skött det speciellt bra hittills, men tyvärr är det nog ganska vanligt inom ett tillverkande företag som har både egna resurser inom systemutveckling och som håller på att lämna det och köpa resurser utanför företaget. I dagsläget har vi ett gäng utvecklare som sitter i USA. Vi har inga kvar i Europa, när det gäller utveckling av kod. Vi har DBA:er (Database Administrators) här i Europa, bland annat en kille som sitter i Ungern och en som ska anställas här i Landskrona. De sköter dels om administrationen av databasen, att man tar back-up, att man *sizar* och att man *re-indexerar* och sådana saker, men de gör även en del utveckling i form av *stored procedures*. Vi är ju en Microsoft-shop, som man hade kunnat kalla oss. Vi kör alltså Microsofts produkter och så har vi SQL Server som vår huvuddatabas.

När det gäller utvecklarna så sitter de som sagt i USA och där har vi tre typer av utveckling som vi gör. Dels är det utveckling i Axapta, vårt affärssystem. Där har vi en duktig kille som sköter om det i USA, men det är bara när vi utvecklar internt. Han har gjort en del av utvecklingen för den europeiska versionen av Axapta. Axapta är nämligen uppdelad i olika instanser och de är tyvärr inte exakt lika [skrattar].

Vi har även en amerikansk instans, som är för Nord- och Sydamerika och där gör han merparten av den utvecklingen. Det görs med en särskild typ av kod som heter *X-kod* och denna kod påminner lite om Java, men är starkt förenklad och har en hel del inbyggt i sig. X-kod är Axaptas programmeringsspråk, kan man säga. Den nyare versionen av Axapta, Microsoft Dynamics AX, är fullt integrerad med Visual Studio och och då kan man programmera i andra språk typ C#.

Ni använder den äldre versionen, alltså Axapta. Finns där några planer på att gå över till Microsoft Dynamics AX?

Ja, absolut. Då går vi över till antingen Dynamics eller till SAP. Jag sa innan att Axapta-utvecklingen enbart var en del av vår utveckling. Den andra utvecklingen som vi gör är att underhålla en del av våra gamla system, som vi benämner som *legacy-applikationer*. Det är ibland strukturerade, men väldigt ofta är det i stil med att någon har utvecklat något och sedan har någon gått förbi och sagt "ja, det där ser bra ut. Det vill jag också ha. Kan inte du paketera det?" och så har det blivit en applikation av det. Vi har tyvärr ganska många sådana. I USA tog man för ganska längesedan ett beslut att man skulle jobba med något som kallas för *Delphi*. Det är ett problem idag, för där finns en hel del Delphi-applikationer kvar och ingen av utvecklarna eller konsulterna kan det längre. Så vi har en back-log här med saker som vi måste konvertera över till något nyare då. Helst ska vi bli av med den, men...

Den tredje utvecklingen som vi gör där är inom webbutveckling, kopplat till vår externa webb, vårt intranät och vårt extranät. Där har vi en utvecklare som gör lite sådan utveckling också, men, och det är ett stort men, så jobbar vi mer och mer med externa partners och vi försöker att använda vår Axapta-utvecklare i USA så lite som möjligt egentligen och i Europa så jobbar vi med en konsultfirma som ligger i Halmstad.

I USA jobbar vi istället med en konsultfirma som ligger i Kansas och dessa får göra allt mer och mer jobb för oss.

Så, hur sköter vi då vår systemutveckling, för det var egentligen det som frågan handlade om. *Vi jobbar inte med agila metoder så mycket utan till väldigt stor del så är det då underhållsprogrammering som sker i dagsläget*. Vi gör väldigt lite nyutveckling. Ingen alls egentligen. Där är vår strategi istället att köpa färdiga system i stället för att utveckla nya.

Den underhållsprogrammeringen vi gör går till på det sättet att vi får in ett förslag på den förändring som ska ske, vilket kan bero på många olika saker. Detta kan vara både i Axapta eller i något av våra gamla system.

Det förslaget kommer först in som kanske ett ärende till vår *helpdesk* och hittills har det varit så att det ärendet har tagits om hand av den som är ansvarig för den applikationen och då har man tillsammans med någon IT-person inom till exempel Axapta försökt hjälpa användaren att dokumentera ner vad det är som denna vill ska förändras och vad som ska göras tydligare.

Vi har en blankett som vi kallar för RFC (*Request for Change*), som innehåller en kortfattad beskrivning om vad man vill göra, vad som händer om man inte gör det (till exempel att det kommer att kosta mer eller att det tar längre tid) och sedan ska man även prioritera hur viktigt detta ärende är (se bilaga 4). Efter att vi har fått en sådan blankett så görs det en grov tidsuppskattning av den och sedan skickas det till företaget i Halmstad. De lägger sedan till ett lösningsförslag och hur många timmar det kommer att ta för dem.

Om vi sedan godkänner detta så blir det just det företaget i Halmstad som utvecklar detta och då följer de sina egna systemutvecklingsmetoder, men vi har en standard inom Axapta som alla måste följa, som beskriver hur man dokumenterar i kod och liknande.

Det var precis det vi tänkte höra om. Hur sköter ni kunskapshanteringen inom dessa projekt?

Alla ändringar som görs måste ha ett RFC-dokument. Det är sedan detta dokument som vi väljer att testa emot när de väl levereras. Ibland görs det en tilläggsdokumentation, men det är bara om det är lite mer komplicerat. Detta görs då i form av en testbeskrivning. I koden däremot så dokumenteras... Varje ärende har ett nummer som skall finnas i koden. Det ska stå vem som har gjort ändringen och man markerar gamla rader med datum när man kommenterar ut dem och sådant. Eftersom affärssystem används av många olika bolag i många olika länder så måste man sätta olika säkerhetsnycklar, så att inte alla får tillgång till detta. Det finns beskrivet lite hur det här ska gå till och vem som ska få lov att köra just den här kodsnutten.

Vi är sämre när det gäller att hantera den när vi väl har skapat den. Den läggs antingen på en *fileshare* någonstans eller så läggs den på några *Sharepoint-sajter*, som tyvärr är ganska interna inom IT-avdelningen. I USA har informationen varit lite mer spridd, men... Vi skickar ut informationen till användarna i samband med testerna. *Tyvärr är det ofta så att dokumentationen är ganska liten*.

För små ändringar är det okej, men det sker inget jobb i att uppdatera någon sorts central dokumentation utan det blir bara ett dokument som beskriver just den lilla ändringen. *Vi har många dokument som finns utspridda på olika ställen och nu i efterhand så är det svårt att hitta tillbaka. Där har vi en stor förbättringspotential*.

Är det mer generell eller detaljerad dokumentation?

Eftersom det är underhållsprogrammering så är det en dokumentation som beskriver de förändringar som man vill göra. *Vi har egentligen ingen grunddokumentation över systemet utan vi använder den som finns från Microsoft*. Där finns hjälptext i systemet som är ganska bra och vi har köpt en användarhandbok som man kan använda, men den är väldigt generell.

Den specifika dokumentationen för Haldex är det sämre med, för vi har nästan "sönderanpassat" systemet. Det finns gammal dokumentation kvar som beskriver de projekt och förändringar som har skett, men den är inte sammanställd.

Finns detta på datorn eller i en pärm?

Om ni ser bakom er [pekar på en bokhylla], så ser ni några pärmar som är märkta med olika länder. De är förstudier som gjordes när man installerade systemet och där man beskrev precis hur verksamhetsprocesserna fungerade och där man försökte skapa någon sorts *gap-analys*, där vi jämförde systemet vi tänkte installera mot verkligheten. Vi försökte då fylla *gapen* eller differenserna med någon sorts programmering.

Eftersom det bara var förstudier så levde dessa enbart under implementeringsfasen och det som sker sedan är någon sorts *förändringsprogrammering*, där RFC-dokumenten spelar roll. Dessa lagras i ett *casesystem*, där vi rapporterar all support och alla förändringar, där varje case har sitt eget nummer.

Till varje case finns det då knutet bilagor, som ofta är ett RFC-dokument, men det är just den samlade dokumentationen som vi saknar. Vi har ju flera tusen sådana här case. Hur ska vi kunna hitta tillbaka till ett specifikt case? Det går att söka på det, men det är enbart grova sökningar då. Vårt caseverktyg är tyvärr ganska gammalt. Vi kan säga så här att om jag inte kan gå in i koden och få någon vägledning där, så är det faktiskt oftare så att jag får kontakta vår leverantör och fråga dem. Då kan man få ledning i koden, för man ser initialer.

Vad heter ert casesystem?

Det heter HEAT (Helpdesk Expert Automation Tool). Det är ett ganska gammalt system. Det fyller sin funktion, men... Till varje ärende som vi har där så skriver vi en *solution description* och ibland är folk duktiga på att skriva in där hur vi har löst ett visst problem eller om det finns en *workaround*.

Tanken är att det ska kunna vara *knowledge based*, men det går inte att använda på ett vettigt sätt.

Vi tror att vi vet svaret redan på nästa fråga, men vi frågar ändå. Använder ni er av en wiki på Haldex?

Nej, det gör vi inte. Tyvärr inte. Det hade varit perfekt att använda för sådana här saker, tycker jag.

Hur definierar du en wiki?

Tja, jag har ingen färdig definition, men jag vill säga att wiki är en plats som man har tillgång till... Jag är inte säker på att den måste vara på nätet, men det underlättar. I alla fall, det är en plats som alla har tillgång till och där man samlar information om ett avgränsat ämne och där man både kan skapa och ändra informationen.

Det finns ju *embryon* till wiki-platser, om man är snäll [skrattar], i form av interna Sharepoint-sidor, där vi säger att här samlar vi information om ändringar. Det är ju skillnad gentemot vårt casesystem, som mer är ett lokalt initiativ.

Sharepoint har ju ett rätt så bra stöd för sökningar...

Jo, men problemet med vår implementering av Sharepoint är ju att vi inte har tagit steget fullt ut och fått ut det i hela organisationen utan det har varit mer eller mindre inofficiella försök ifrån IT-avdelningen att starta olika saker. Nu har vi projekt inom de här områdena så jag hoppas att vi kommer igång med de här grejerna. Tyvärr har vi inte hunnit fixa till de här sakerna.

Har ni använt er av en wiki tidigare?

Inte här på Haldex i alla fall. Jag har jobbat lite med det tidigare och jag har goda erfarenheter av det. *Så länge ämnet är något som engagerar folk så lever ju wikin. Det är inte något som du bara kan starta upp och tro att det kan underhålla sig själv utan du måste underhålla och föda det med information*. Det är inte något som gör att man kan svara på alla frågor, men...

Finns där några officiella planer på att implementera en wiki på Haldex?

Jag får säga att jag fick faktiskt liknande frågor av min fru nu i helgen. Hon hade varit på en konferens för lärare, för hon är lärare, och där hade de pratat om wiki och då skapade hon sin egen wiki för att använda för sitt jobb. Hon frågade mig då om vi hade det här på Haldex och då sa jag "njä, det har vi inte, men det är ingen dum idé faktiskt". Det är ingen dum idé alls. Vi har precis identifierat olika globala processägare inom Haldex och nu ska de här processägarna då skapa en organisation och definiera hur deras processer ser ut. Man hade faktiskt kunnat tänka sig att man hade en wiki per huvudprocess där de som är involverade i processen skulle kunna dela med sig av information, för det är ett snabbt sätt att strukturera information. *Det finns information på många ställen, men den är inte strukturerad och den är inte samlad. En wiki skulle kunna vara ett sätt att lösa det på faktiskt.*

Vi har också en ny *ICT-sajt*, som vi kallar det för på Sharepoint, där vi hoppas att den ska kunna leva vidare. Samtidigt har vi en del gamla ställen där vi har gammal information, där vi ser på vad vi ska göra med den och om vi ska behålla den eller inte. Då hade en wiki nog kunnat vara ett bra sätt att lagra den informationen på så att den blir åtkomlig. För att ta ett konkret exempel: vi hade ett projekt där vi ville implementera Axapta i Indien för flera år sedan. Projektet blev aldrig av, men nu börjar det snackas om att det kanske ska försökas på igen. Då kom frågan var vi hade den gamla informationen lagrad. Jag hade tur att det var någon som fanns med från den tiden som visste var den gamla informationen fanns.

I annat fall kanske jag hade varit tvungen att göra om allt arbete som vi redan har gjort tidigare, om jag inte hade fått reda på att där fanns gammal information redan.

Då ska vi se om vi har förstått dig rätt. Det finns alltså olika verktyg på IT-avdelningen, då bland annat Sharepoint, ert casesystem HEAT och en fileshare, men även pärmar. Är det de verktyg som ni använder?

Ja, det är det. Fast för att driva själva systemutvecklingen så är det RFC-dokumentet som är viktigt, som är ett Word-dokument i princip. Användaren är den som fyller på det först och sedan fyller vi på det med mer information innan det skickas till en systemutvecklare, som antingen är intern eller extern, som skickar tillbaka sitt lösningsförslag och sin tidsuppskattning. Sedan testas detta och *fajlas*, men där har vi inget bra ställe, tyvärr.

Finns det ingen central plats där det samlas?

Det finns, men... En hel del av systemutvecklingsdokumenten hamnar inte där, men det finns *dokumentbibliotek*. Det finns ett system som heter WebDocs som vi använder här i Landskrona, där ledningen har processbeskrivningar. Jag vet att en del av de dokument som beskriver de olika IT-systemen finns där, men jag gick in där för ett tag sedan och såg att det fanns dokument som beskrev hur man arbetade i Oracle, som vi hade innan Axapta. Om Axapta är gammalt så är Oracle urgammalt [skratt]. Där finns alltså ett problem med att hålla detta uppdaterat. WebDocs är anpassat efter Haldex Landskrona, men eftersom system ofta är globala så behöver vi något som funkar för alla enheter.

Vi ska inte underskatta en stor och intensiv mailkonversation också [skrattar]. Det är ju så att många av diskussionerna runt förändringarna kommer i form av mail. Det är ju klart att några av dessa mail är *nyckelmail*, som förklarar syftet med varför man gjorde på ett visst sätt.

Men om vi säger att du behöver någon sorts information, hur går du då tillväga för att få tag på denna?

Om jag behöver veta hur ett system fungerar så letar jag först på någon av de *fileshares* som vi har. Annars letar jag på någon av de Sharepoint-sajter som vi har. Jag tittar naturligtvis i min mail. Troligtvis har jag fått någon information om detta innan, så det kommer inte som en blixt ifrån en klar himmel. Dock är det så att jag kan få frågor om fält som gör att jag måste gå in i koden och undersöka om det är ett standardfält eller om det är något som vi har lagt till. Detta kan vara ganska omfattande undersökningar.

Många gånger kan man också gå till verksamheten, alltså de som använder det här. De vet ju ofta en del om man bara hittar rätt person att fråga.

Finns det något globalt samarbetssystem mellan era kontor?

Inte mer än att vi har samma nätverk, att vi har samma mail och att vi använder Office Communicator. De har även tillgång till vår fileshare, men de har begränsad behörighet. Det finns en del, men vi håller på att göra det mer globalt. Det känns som att i USA har man sina delar, i Europa har man sina delar och i Kina har man sina delar... Vi har varit ganska dåliga på att dela med oss av informationen till de olika regionerna.

Sedan är det så att allt inte är på engelska, så det är något som vi får se till att bli bättre på. Detta påverkar också hur pass tillgänglig informationen är.

Företaget i Halmstad, har de också ett arkiv med Haldex-dokument?

Ja, det har de. Jag vet att de har informationen på en intern fileshare, men även i pärmar. Ägaren till företaget, som är vår kontaktperson, har även *mappar* i sin Outlook där han samlar information, så om jag mot förmodan skulle ha missat att *fajla* något av dessa mail så har han säkert gjort det, så det går att hitta.

Och om han byter jobb eller måste rensa sin Outlook...?

Ja, så är det. Då har vi ibland lämnat över dessa .pst-filer [Personal Storage Table, mailarkiv-filer i Outlook], men det är alltid svårt att hitta i någon annans struktur. Så man tvekar inför detta varje gång. Informationen används därför inte så mycket eftersom den är svår att söka i, men om jag måste så går det ju fast det tar längre tid.

Tror du att valet av systemutvecklingsmetod påverkar sättet att dokumentera?

Ja, det gör det. Valet av systemutvecklingsmetod är en konsekvens av den typen av utveckling vi gör, vilket jag kallar för förändringsutveckling snarare än nyutveckling. Hade vi gjort nyutveckling så hade vi förmodligen haft ett större behov av fler dokument. Nu är processen och systemet känt för användarna som vi har diskussion med.

Det vi egentligen vill förmedla och dokumentera är de förändringar vi vill göra. Då är ett vanligt Word-dokument ganska bra faktiskt. Där finns det med skärmdumpar, bilder och figurer som visar hur det ska fungera.

Det är sällan något blir så stort att det måste modelleras. Det täcker ganska väl våra behov och det är en konsekvens av den typen av programmering vi gör. Det jag tror att vi saknar och det vi behöver är kanske en tydligare *spårbarhet*. Det är någonting som vi ibland får kritik för från revisorer och annat. Det ska vara tydligare att se, vilken ändring har orsakat vilka kodrader, så att man får spårbarheten ända ner i koden, egentligen.

Det finns ett tekniskt problem i form av att alla verktygen inklusive Axapta är lite åldersdigna. Det är svårt att jobba med det. Man har inget stöd för *incheckning* av kod. Det går inte riktigt att ha ett modernt arbetssätt med de här gamla systemen, men förhoppningsvis med nyare system och nyare grejer här så kommer vi få till det också. Det har också lite med det att göra att man måste anpassa systemutvecklingen till den verklighet som man befinner sig i också. Det är på gott och ont, men naturligtvis är det också så att det funnits en ovilja att betala för mycket för dokumentation. Vi har ju flera gånger sagt till våra partners och konsulter att ju mindre administration desto bättre och att vi måste prioritera för att göra våra ändringar. Fast så klart, någon form av miniminivå måste vi ha.

Haldex är ju inget litet företag. Det kan ta tid för att implementera ett nytt system.

Precis. Det påverkar också. Jag skulle vilja säga så här: inom vissa områden har vi ju vita fläckar. Där går det ibland snabbt att implementera nytt där det inte finns något att ersätta.

För få in, exempelvis, Sharepoint inom vissa delar, med eller utan wiki, så går det ju ganska bra, faktiskt. Inom vissa områden kan man få gehör ganska snabbt. Om det finns ett behov och ett verktyg för det, då kan man verkligen använda det.

[...]

[Visar RFC-dokument i pärmen] Man skulle kunna titta på ankomstkontroll, till exempel. Då har man skrivit upp alla närvarande, alltså de som var på det här mötet. Nu var detta före min tid. Då så har man beskrivit, hur processen skulle gå till, om det finns några *statusar*, vad det är för saker som man vill kontrollera. Så man beskriver processen ganska detaljerat och gör noteringar. Här är någon Magnus som har sagt att den här processen inte gäller för "Luft", som tydligen är annorlunda då och här så småningom har man önskemål för framtida ändringar där man diskuterar att man kanske hellre skulle vilja så här. I framtida processer vill man att det funkar på det sättet i stället alltså.

Det är intressant. Vi var på ett företag som heter Syncron och de har också den typen av ärenden fast i en wiki.

Ja, precis. Det här kunde man lika gärna satt in i en wiki. Fast det här är en förstudie och det är ganska käckt att ha det samlat i ett dokument. För att ofta är det så här att man refererar till det här dokumentet i samband med en upphandling sedan och säger att det är så här det ska fungera då.

Man skiljer på vanlig wiki såsom Wikipedia till exempel och enterprise wiki, som kan integreras med ärendehanteringen. Varje ärende får ett nummer som man referera till också.

[Bläddrar i pärmen] I våra RFC-dokument finns ID, namn på de som vill ha det och var inom Haldex vi vill göra detta då. Vi hade ju tre divisioner, nu har vi bara en, vilket bolag det är som påverkas av det, är det bara Landskrona eller Europa eller hela världen, och vilken prioritet. Det brukar inte vara smart att skriva "low" här, för att man brukar inte få igenom det då [skrattar]. Här skriver man då vad man vill göra, vad det är för konsekvenser om man inte gör det, en *overview over proposed solution*. När det här kommer iväg, så brukar företaget i Halmstad i efterhand här skriva hur de kan lösa det och hur mycket tid det skulle ta.

Så skickar de det till dig sedan...

Ja. Sedan får jag stämma av det. Säg, om det vill ha 13 timmar för att göra det. Jag avväger om det är vettigt och ser om vi ska beställa detta eller inte. Beroende på hur stort det och hur många människor är med tar vi beslutet då. Om vi väljer att göra det svarar jag till företaget i Halmstad "var snäll och genomför detta och berätta för mig när det är klart". När de är klara med det testar vi det. Så kommer det med i nästa *release* så småningom. Request for Change är ett av de viktigaste dokumenten vi har. Nu kommer vi ändra lite processer här framåt, men i princip så är det så att så vi kommer att jobba även framöver. Hoppas det blir mer uppstyrt med dokumentationen. Det är viktigt att få ordning på den. Det kan vara så att jag fick en idé här med wiki.

Tack så mycket för intervjun. Det var snällt att du ville ställa upp.

Vad bra, då hoppas jag att ni fått svar på era frågor. Om det är någonting ni saknar, så är det bara att höra av sig.

Bilaga 4 - RFC-dokument (Request for Change)

Haldex

Request for Change IS/IT Reg. No Page

Issued by Date Issue

Functional Specification

HEAT ID	22431
Sub-category	[ERP]
Requester	[redacted]
Email	[redacted]
Phone	[redacted]
Division	[redacted]
Business area/Company	[redacted]
Location	Weyersheim and Heidelberg
Issue date	2010-08-06
Priority [high/medium/low]	high
Cost center	
Sign off by	

1.0	**Statement of Issue** *Description of What is the issue and Why is a customization necessary. Describe Requirements and goals.*
	The goods receiving team often book the goods against the wrong delivery schedule line meaning not against the oldest one because the delivery schedule lines are not sorted on delivery date and/or promised date

2.0	**Consequence if not implemented?** *Describe the urgency of the problem.*
	Wrong lines are received and lines that were scheduled to be received are shown as still open and therefore expected. It means also that all the dates must be reworked in the delivery schedule, also the delivery reminders and activities and corrupted and so on…

3.0	**Overview of Proposed Solution** *How is it suggested that specific requirements and goals are met.* *Included in the description should be a descritption of assumptions, constraints, procedures etc.* *How is the added functionality implemented to the solution?* *Where in the Menu should the change be added? Does a work around exist?*
	The screen must automatically propose the correct lines meaning take at first the oldest delivery date and if there are several lines with the same delivery date then take the oldest promised date at first and if both are the same then take the lowest line number at first

4.0	**Benefits & Business Case** *Description of benefits for Haldex – economic and organizational.*
	Benefits would to not destruct the work of the procurement team trying to put the correct delivery dates and promised dates in the system to enable sales and production to rely on the dates they see in the net requirement screen. Elimination of waste also due to not redo several the same work meaning correct the wrong received delivery schedules

Form No. 20074771/2008-10-09/PerEriksson Ute. 1 HALDEX BRAKE PRODUCTS AB

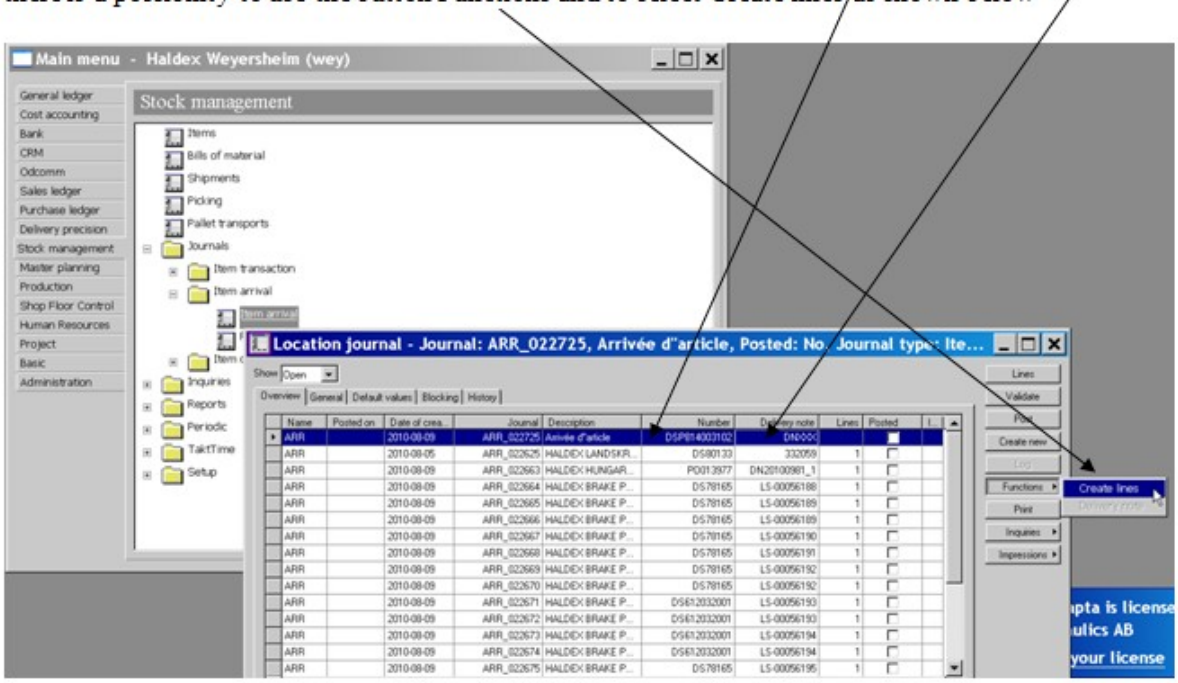
Haldex

Request for Change IS/IT Reg. No Page

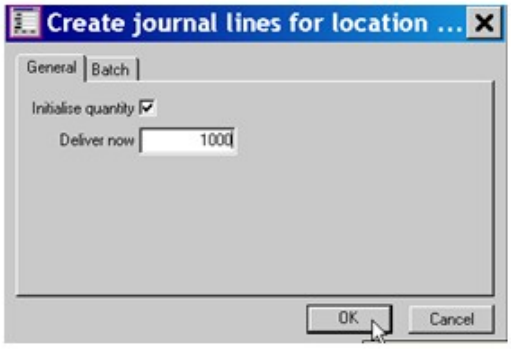
Issued by Date Issue

5.0	**Detailed Requirements & Specifications** *Layout examples e.g.:* • *Scan Lists (Electronic form)* • *Screen Dumps with Changes* • *How to test, what should be the result etc.*

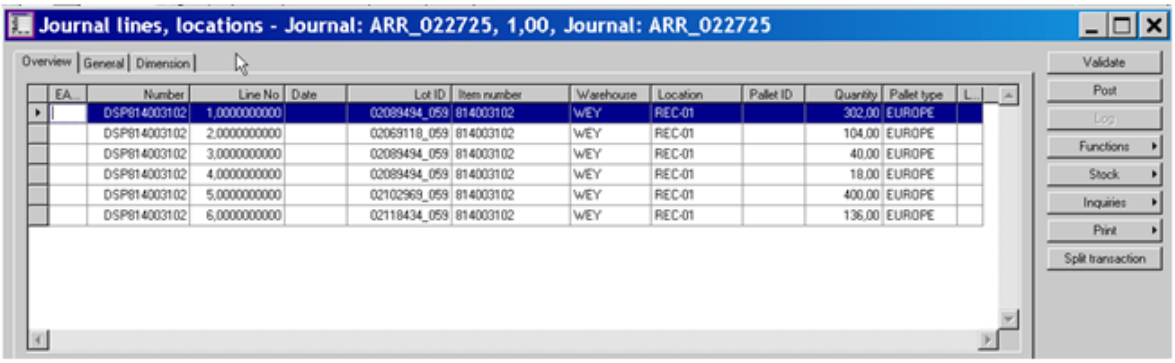
In the item arrival journal screen, once the header is initiated with the Number and the Delivery note, there is a possibility to use the button Functions and to select Create lines as shown below



You can then enter the total quantity you want to receive and cross the box Initialise quantity



The result is as below, as you can see Axapta initialize well according to the quantity of 1000pcs. But the issue here is that Axapta initialize the several delivery schedule lines against the Line No or Lot ID, I'm not really sure which of them is used. But it's sure that Axapta does not initialize against the delivery dates at first and then the confirmed dates at second.



EA...	Number	Line No	Date	Lot ID	Item number	Warehouse	Location	Pallet ID	Quantity	Pallet type	L...
	DSP8140031 02	1,0000000000		02089494_059	814003102	WEY	REC-01		302,00	EUROPE	
	DSP8140031 02	2,0000000000		02069118_059	814003102	WEY	REC-01		104,00	EUROPE	
	DSP8140031 02	3,0000000000		02089494_059	814003102	WEY	REC-01		40,00	EUROPE	
	DSP8140031 02	4,0000000000		02089494_059	814003102	WEY	REC-01		18,00	EUROPE	
	DSP8140031 02	5,0000000000		02102969_059	814003102	WEY	REC-01		400,00	EUROPE	
	DSP8140031 02	6,0000000000		02118434_059	814003102	WEY	REC-01		136,00	EUROPE	

The change should ensure that the 1000pcs initialized correspond to the 1000pcs that have the earliest delivery dates and if several lines concerned have the same delivery date then it should initialize against promised date if some are listed. If several lines concerned have the same delivery date and promised date then it should take the lowest line number. And this for both Haldex Weyersheim and Heidelberg.

Thanks and best regards

Bilaga 5 - Organisationsinformation

Tretton37 AB

Tretton37 är ett relativt litet konsultföretag i Lund, startat 2010, som har specialiserat sig på .NET-utveckling. Tretton37 har en platt organisationsstruktur. I skrivande stund är det 25 konsulter som i denna kunskapsorienterade organisation kallar sig för *ninjas*. Tretton37:s verksamhet kretsar runt att hyra ut seniora systemutvecklare till lokala företag, det vill säga att de hyr ut konsulter. Trots att det inte finns någon gemensam *in-house*-utveckling försöker företaget att hålla ihop sina anställda genom att organisera kvällsmöten såsom *code nights*, *code camps*, teknikfrukostar och mycket mer, där kunskapsutbyte står i centrum. Företaget värdesätter kunskapsutbytet även externt genom att genomföra seminarier, diskussionskvällar och de deltar även aktivt på sociala medier därför att de anser att man inte bara ska ta utan även ge. Tretton37 har ingen företagswiki, men däremot används wikiverktyg av konsulterna (“ninjorna”) när de är ute på uppdrag hos andra organisationer.

Syncron AB

Syncron utvecklar produkthanteringssystem i Java, där de även installerar, konfigurerar och underhåller dem hos industriorganisationer världen över. Syncron har en hierarkisk organisationsstruktur. Syncrons huvudkontor ligger i Malmö, men de flesta systemutvecklare är på kontoret i Warszawa. Utöver dessa två kontor har Syncron även kontor i bland annat Tokyo och London. På grund av spridningen av medarbetare är Syncron beroende av ett välfungerande dokumentationssystem. Hjärtat i företaget är den programvara företaget utvecklar och support av denna. Versionshanteringssystemet och ärendehanteringssystemet för både kod och support är kopplade till den interna *enterprise wikin*, som har använts sedan tre år tillbaka. “Ingenting görs förrän man skriver in detta i wikin eller i ärendehanteringssystemet” sa den personen som vi intervjuade därifrån. Detta är för att alla involverade, som befinner sig på olika kontinenter och tidszoner, ska kunna ta del av hur saker och ting går.

Bool Nordic AB

Bool är ett nischat företag som har specialiserat sig på utveckling av Sharepoint-baserade portaler. Bool har en platt organisationsstruktur. Företaget har cirka 25 anställda, både in-house och konsulter, och ligger i Lund. Deras kunder återfinns mestadels i Skåne-regionen, men de har även kunder i Stockholm och Norge. Företaget har en egendesignad systemutvecklingsmetodik som grundar sig på agila principer och lean-filosofin. Systemutvecklingsmetodiken kallas för Boolean och använder sig av bland annat en kanban-tavla i fysisk form. Boolean är speciellt anpassad för Bool och dess storlek, specialisering på Sharepoint och projektmönster där en av Booleans principer är att metoden ska utveckla sig själv kontinuerligt. Bool fokuserar på direkt kommunikation mellan projektdeltagarna och medarbetare och varje projekt har en egen sajt i Bools intranät, som bygger på Sharepoint.

Bool använder även en wiki, som också är Sharepoint-baserad, för generell information om utveckling och den egna infrastrukturen. Tanken är att så fort någon har utforskat något nytt så ska det skrivas upp på wikin. Det finns inga krav på struktur i wikin på Bool, utan det viktigaste är att det ska gå snabbt att skriva in det.

Haldex AB

Haldex är ett multinationellt företag som producerar bildelar. Haldex har en hierarkisk organisationsstruktur. Efter omstruktureringar och försäljning av vissa produktioner sysslar Haldex i Landskrona mest med *Commercial Vehicle Systems* (CVS) och bromssystem. Haldex affärssystem är baserat på Axapta, en produkt av Microsoft, som var föregångaren till det som idag kallas för Dynamics AX. Axapta har blivit anpassat mycket just för Haldex och alla ändringar dokumenteras i form av speciella blanketter som heter *Request for Change*, som fylls i av både beställaren och utvecklaren. All dokumentation sparas sedan i fysiska pärmar, men även i Outlook-mappar (i så kallade .pst-filer). Det är dessa filer som sedan lämnas över vid ett eventuellt jobbyte, för att ge efterträdaren chans att ta del av nödvändig information.

Haldex har även en Sharepoint-baserad sajt som används internt av ICT-avdelningen (Information and Communications Technology). Haldex har numera inte särskilt många systemutvecklare utan man försöker lägga ut systemutvecklingen till andra företag. Sedan tidigare finns det många program som är skrivna i programspråket Delphi och den personen som vi intervjuade där kände att detta språk var förlegat.

Referenser

Alajbegovic, A., Olofsson, H., Scholten, M. (2011): *Low-techverktyg i high-techmiljöer.* C-uppsats. Institutionen för informatik, Lunds universitet, Lund.

Alavi, M., Leidner, D. E. (1999): Knowledge Management Systems: Issues, Challenges, and Benefits. *Communications of the AIS* (vol. 1, nr. 1), sid. 1-37.

Alavi, M., Leidner, D. E. (2001): Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues. *MIS Quarterly* (vol. 25, nr. 1), sid. 107-136.

Andersson, L. (2010): *Knowledge Management System och kollaborativt arbete - Wiki som informationshanteringsverktyg*. C-uppsats. Centrum för teknikstudier, Malmö högskola, Malmö.

Andersson, O., Rollof, O. (2010): *Kunskapshantering med interna Wiki: Hur kunskap skapas och överförs i organisationer.* C-uppsats. Institutionen för informatik, Lunds universitet, Lund.

Aubel, J. (1994): *Guidelines for Studies Using the Group Interview Technique*. International Labour Organization, Genève, Schweiz.

Boden, A., Avram, G., Bannon, L., Wulf, V. (2009): Knowledge Management in Distributed Software Development Teams - Does Culture Matter?. *4th IEEE International Conference on Global Software Engineering,* IEEE Publisher, Limerick, Irland.

Burton, J. K. (2005): *Information Sharing and Collaboration Business Plan*. D-3206, Institute for Defense Analyses, Alexandria, Virginia, USA.

Carr, D. F. (2007): The Enterprise Wiki. *Baseline* (vol. 72), sid. 84.

Chatti, M. A., Klamma, R., Jarke, M., Naeve, A. (2007): The Web 2.0 Driven SECI Model Based Learning Process. *7th IEEE International Conference on Advanced Learning Technologies*, Niigata, Japan.

Chau, T., Maurer, F. (2005): A Case Study of Wiki-based Experience Repository at a Medium-sized Software Company. *3rd International Conference on Knowledge Capture*, Banff, Alberta, Kanada.

Cormode, G., Krishnamurthy, B. (2008): Key Differences Between Web 1.0 and Web 2.0. *First Monday* (vol. 13, nr. 6).

Cross, R., Parker, A., Borgatti, S. P. (2002): *A Bird's-eye View: Using Social Network Analysis to Improve Knowledge Creation and Sharing*. IBM Institute for Business Value, Somers, New York, USA.

Davenport, T. H., Prusak, L. (1998): *Working Knowledge: How Organizations Manage What They Know*. Harvard Business School Press, Boston, Massachusetts, USA.

Decker, B., Ras, E., Rech, J., Jaubert, P., Rieth, M. (2007): Wiki-based Stakeholder Participation in Requirements Engineering. *IEEE Software* (mars-april 2007), sid. 28-35.

Dey, I. (1993): *Qualitative Data Analysis: A User-friendly Guide for Social Scientists*. Biddles Ltd., Guildford och King's Lynn, Storbritannien.

Dingsøyr, T., Dybå, T., Moe, N. B. (2010): Agile Software Development: An Introduction and Overview. Dingsøyr, T., Dybå, T., Moe, N. B. red. (2010): *Agile Software Development: Current Research and Future Directions*, Springer, Berlin, Tyskland.

Drucker, P. F. (1991): The New Productivity Challenge. *Harvard Business Review* (november-december 1991), sid. 69-79.

Fahey, L., Prusak, L. (1998): The Eleven Deadliest Sins of Knowledge Management. *California Management Review* (vol. 40, nr. 3), sid. 265-276.

Gonzalez-Reinhart, J. (2005): Wiki and the Wiki Way: Beyond a Knowledge Management Solution. *Information Systems Research Center*, sid. 1-22.

Hargreaves, A. (2002): Teaching in the Knowledge Society. *Vision 2020 - Second International Online Conference*, Technology Colleges Trust, Boston College, Massachusetts, USA.

Hollingsworth, C. (2011): What Kanban Can Do. *PM Network* (vol. 25, nr. 3), sid. 66-67.

Jacobsen, D. I. (2009): *Vad, hur och varför?: Om metodval i företagsekonomi och andra samhällsvetenskapliga ämnen*. Studentlitteratur, Lund, Sverige.

Jenefeldt, J. (2010): *Kunskapsspridning med ett wiki: Är wiki ett effektivt verktyg för att sprida kunskap i ett företag?*. C-uppsats. Institutionen för datavetenskap, fysik och matematik, Linnéuniversitetet, Kalmar.

Jing, H., Fan, Y. (2008): Usability of Wiki for Knowledge Management of Knowledge-based Enterprises. *2008 International Symposium on Knowledge Acquisition and Modeling*, IEEE Computer Society Press, Los Alamitos, Kalifornien, USA.

Kaplan, A. (2010): *Att designa ett multiplattformsystem: Interaktionsdesign möter agila projektmetoder.* C-uppsats. Institutionen för datavetenskap, Linköpings universitet, Linköping.

Kimmerle, J., Cress, U., Held, C. (2010): The Interplay Between Individual and Collective Knowledge: Technologies for Organisational Learning and Knowledge Building. *Knowledge Management Research & Practice* (vol. 8), sid. 33-44.

Leander, O. (2010): *Förändring av produktionslayout: Studier av omfattande förändringar i produktionslayouten*. C-uppsats. Institutionen Ingenjörshögskolan, Högskolan i Borås, Borås.

Long, S. A. (2006): Exploring the Wiki World: The New Face of Collaboration. *New Library World* (vol. 107, nr. 1222/1223), sid. 157-159.

Lytras, M. D., Damiani, E., Ordóñes de Pablos, P. (2009): *Web 2.0: The Business Model*. Springer Science + Business Media, New York, New York, USA.

Majchrzak, A., Wagner, C., Yates, D. (2006): Corporate Wiki Users: Results of a Survey. *2006 International Symposium on Wikis*, Odense, Danmark.

Mirza, Z. H. (2010): *Illustrating the Use of Agile Software Development Process*. C-uppsats. Tillämpad informationsteknologi, Göteborgs universitet, Göteborg.

Moratilla Temprado, E., Ruz Bendito, E. (2010): *Lean Software Development and Agile Methodologies for a Small Software Development Organization*. D-uppsats. Institutionen Ingenjörshögskolan, Högskolan i Borås, Borås.

Müller, C., Meuthrath, B., Baumgraß, A. (2008): Analyzing Wiki-based Networks to Improve Knowledge Processes in Organizations. *Journal of Universal Computer Science* (vol. 14, nr. 4), sid. 526-545.

Newman, B. D., Conrad, K. W. (2000): A Framework for Characterizing Knowledge Management Methods, Practices, and Technologies. *Proceedings of the Third International Conference on Practical Aspects of Knowledge Management (PAKM2000),* CEUR Workshop Proceedings, Basel, Schweiz.

Nonaka, I. (1994): A Dynamic Theory of Organizational Knowledge Creation. *Organization Science* (vol. 5, nr. 1), sid. 14-37.

Nonaka, I., Konno, N. (1998): The Concept of "Ba": Building a Foundation for Knowledge Creation. *California Management Review* (vol. 40, nr. 3), sid 40-54.

Nordin, M., Öhlin, D. (2009): *Skapa en webbplats med Mediawiki: En guide om hur man använder publiceringsverktyget Mediawiki*. Stiftelsen för Internetinfrastruktur (.SE), Stockholm, Sverige.

Oates, B. J. (2006): *Researching Information Systems and Computing*. Sage Publications, London, Storbritannien.

Olsson, M. (2008): Wikin - vägen till enterprise 2.0. *Internetworld* (läst den 4 maj 2011).

O'Reilly, T. (2007): What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software. *Communications & Strategies* (nr. 65), sid. 17-37.

Patton, M. Q. (1987): *How to Use Qualitative Methods in Evaluation*. Sage Publications, Newbury Park, Kalifornien, USA.

Pettersson, F. (2007): 10 fördelar med wikis. *WikiSverige* (läst den 4 maj 2011).

Ruparelia, N.B. (2010): *Software Development Lifecycle Models*. ACM SIGSOFT Software Engineering Notes, (vol. 35, nr. 3).

Sieloff, C. G. (1999): "If only HP knew what HP knows": The Roots of Knowledge Management at Hewlett-Packard. *Journal of Knowledge Management* (vol. 3, nr. 1), sid. 47-53.

Spraggon, M., Bodolica, V. (2008): Knowledge Creation Processes in Small Innovative Hi-tech Firms. *Management Research News* (vol. 31, nr. 11), sid. 879-894.