

# Scrum med XP-relaterade tekniker

Införandet av Scrum och dess  
påverkan på systemutvecklargruppen

Magisteruppsats, 15 högskolepoäng, INFM01 i Informatik

*Framlagd:* 06, 2008

*Författare:* Erik Forsberg  
Martin Lindström

*Handledare:* Pär-Ola Zander

*Examinatorer:* Odd Steen, Erik Wallin

<b>Titel</b>	Scrum med XP-relaterade tekniker, Införandet av Scrum och dess påverkan på systemutvecklargruppen.
<b>Författare</b>	Erik Forsberg Martin Lindström
<b>Utgivare</b>	Institutionen för informatik
<b>Handledare</b>	Pär-Ola Zander
<b>Examinatorer</b>	Odd Steen Erik Wallin
<b>Publiceringsår</b>	2008
<b>Uppsatstyp</b>	Magisteruppsats
<b>Språk</b>	Svenska
<b>Nyckelord</b>	Agile, Scrum, Scrum Team, gruppsykologi

### **Abstract**

Scrum är ett agilt ramverk för projekthantering, tänkt att effektivisera systemutvecklingsarbete och maximera affärsnyttan, bland annat genom användandet av roller, så som en icke detaljstyrande projektledare (ScrumMaster), en engagerad produktägare (Product Owner), och en självständig systemutvecklargrupp med ett brett spektrum av färdigheter (Scrum Team). I följande uppsats ämnar vi att undersöka hur ett Scrum Team och deras arbete blir påverkade av att vara delaktiga i en Scrumimplementering. För att åstadkomma detta har vi genomfört en fallstudie med kvalitativa intervjuer, direktobservationer i arbetsmiljö samt insamlande av dokument. Förutom dessa källor har vi genomfört en expertintervju med en av grundarna bakom Scrum, Jeff Sutherland. Våra fynd tyder på att förändringarna inkluderar en rad faktorer, så som samarbete, transparens och kommunikation, men framför allt en ökning i självbestämmande och självständighet, samt en ökning av kollektivt ansvarstagande. Bland de negativa effekterna återfinns bland annat en brist på översikt.

## Innehållsförteckning

Innehållsförteckning .....	1
1 Inledning.....	4
1.1 Bakgrund .....	4
1.2 Forskningsfråga .....	7
1.3 Syfte.....	7
1.4 Målgrupp .....	7
1.5 Avgränsningar .....	7
1.6 Disposition.....	8
2 Teoretisk bakgrund.....	9
2.1 Agil systemutveckling kontra sekventiell .....	9
2.2 Agila tekniker och Extreme Programming (XP).....	14
2.2.1 Unit Test och Test Driven Development.....	14
2.2.2 Coding Standards Coding Guidelines .....	14
2.2.3 Code Review .....	15
2.2.4 User Stories och Use Cases .....	15
2.2.5 Kollektivt ägande av kod.....	15
2.2.6 Continuous Integration .....	15
2.3 Scrum.....	16
2.3.1 Roller i Scrum .....	18
2.3.2 Ceremonier i Scrum.....	20
2.3.3 Artefakter i Scrum .....	20
2.3.4 Arbetsprocessen i Scrum .....	22
2.4 Problematik kopplat till att arbeta agilt .....	24
2.5 Psykologi, grupper och gruppdynamik .....	26
2.5.1 Grupp effektivitet .....	27
2.6 Gruppsykologiska faktorer och dess inverkan på effektivitet .....	28
2.6.1 Ledarskap .....	28
2.6.2 Kommunikation.....	28
2.6.3 Beslutsfattande .....	29
2.6.4 Konflikter .....	30
2.6.5 Sammansättning .....	30
2.7 Externa faktorer .....	31
2.7.1 Organisatorisk kontext .....	31
2.7.2 Kulturell kontext.....	31
2.8 Att sätta samman en grupp .....	32
2.9 Konceptuell modell .....	33
2.9.1 Faktorerna som påverkar Scrum Team .....	33
2.9.2 Organisationens eventuella påtryckningar på Scrum Team .....	34
2.9.3 Organisationen och ScrumMasters påverkan på grupp sammansättningen .....	35
2.9.4 Product Owner och dennes koppling till teamet .....	35
2.9.5 ScrumMasters roll i Scrumprocessen, och kopplingen till Scrum Team .....	36
2.9.6 Scrumprocessens arv från det Agila tänkandet .....	37

2.9.7 Teamet och Scrumprocessen .....	37
3 Metod.....	38
3.1 Forskningsstrategi .....	38
3.2 Konceptuell modell .....	40
3.3 Urval.....	41
3.4 Datainsamling.....	41
3.4.1 Förberedelser för insamling av data .....	41
3.4.2 Insamling av data.....	41
3.4.3 Negativa instanser .....	42
3.5 Analys av data .....	43
3.6 Etik och forskningskvalitet.....	44
3.6.1 Övergripande etiska ställningstaganden .....	44
3.6.2 Reliabilitet .....	45
3.6.3 Validitet .....	45
3.6.4 Bias och dess hantering .....	46
3.6.5 Generalisering .....	47
3.6.6 Källkritik .....	47
4 Resultat och analys.....	48
4.1 Fallbeskrivning .....	48
4.1.1 Avdelningen och respondenter .....	48
4.1.2 Scrum på avdelningen Test Automation Development.....	49
4.2 Utfall.....	50
4.2.1 Organisation .....	50
4.2.2 ScrumMaster .....	51
4.2.3 Product Owner.....	56
4.2.4 Scrum Team .....	59
4.2.5 Scrum Team, gruppsykologiska faktorer .....	60
4.2.6 Scrum Team, individuella faktorer.....	65
4.2.7 Scrumprocessen.....	69
4.2.8 Agile och XP .....	73
4.3 Sammanfattning.....	74
4.3.1 Organisation .....	74
4.3.2 ScrumMaster .....	74
4.3.3 Product Owner.....	75
4.3.4 Scrumprocessen.....	76
4.3.5 Scrum Team, gruppsykologiska faktorer .....	76
4.3.6 Scrum Team, individuella faktorer.....	77
4.3.7 Agile och XP .....	77
5 Expertintervju, Jeff Sutherland.....	78
6 Diskussion och slutsatser.....	81
6.1 Diskussion .....	81
6.2 Hur påverkas systemutvecklargruppens arbete? .....	81
6.2.1 Självbestämmande och självorganiserande .....	81
6.2.2 Kollektivt ansvarstagande .....	82
6.2.3 Kommunikation och öppenhet .....	82
6.2.4 Samarbete .....	83
6.2.5 Transparens .....	83
6.2.6 Agila tekniker som stöd i arbetet.....	84
6.3 Vilka faktorer påverkar resultatet? .....	84
6.3.1 Utbildning och processkunskap.....	84

6.3.2 ScrumMaster skyddar ScrumTeam .....	85
6.3.3 Sammansättning av ScrumTeam .....	86
6.3.4 Organisatoriskt stöd.....	87
6.3.5 Tidigare erfarenheter och yrkesskicklighet .....	87
6.4 Slutsatser .....	88
6.5 Vidare forskning.....	89
Referenslista .....	90
<b>Bilagor</b>	
Bilaga 1, Fallstudieprotokoll	
Bilaga 2, Agile Manifesto And Principles	
Bilaga 3, Mall över kodningskategorier	
Bilaga 4, Exempel på kodning	
Bilaga 5, Direktobservation: Daily Scrum nr 1	
Bilaga 6, Direktobservation Daily Scrum nr 2	
Bilaga 7, Observationer i arbetsmiljön	
Bilaga 8, Utdrag av Sprint Backlog	
Bilaga 9, Burndown Charts	
Bilaga 10, Anders Nyberg, Linjechef	
Bilaga 11, Lars Ahlkvist, ScrumMaster	
Bilaga 12, Niklas Holmberg, Product Owner	
Bilaga 13, David Jönsson, medlem i Scrum Team	
Bilaga 14, Martin Sternebring, medlem i Scrum Team	
Bilaga 15, Karl Eklund, medlem i Scrum Team	
Bilaga 16, Jonas Nyberg, medlem i Scrum Team	
Bilaga 17, Exempel på Unit Test	
Bilaga 18, Exempel på Use Case	
Bilaga 19, Exempel på User Story	

# 1 Inledning

---

*Följande kapitel agerar som en introduktion till området, och definierar även syftet med uppsatsen, samt dess forskningsfråga, målgrupp och avgränsningar.*

---

## 1.1 Bakgrund

Systemutvecklingsprojekt adresserar ofta komplexa problem, för att hantera dessa problem finns det i dagsläget en uppsjö av olika systemutvecklingsmetoder, vilka säger sig kunna öka produktiviteten och kvaliteten på det som produceras (Fitzgerald et al 2002). Komplexiteten består enligt Schwaber (1995) av en kombination av miljövariabler i den aktuella systemutvecklingsmiljön och miljövariabler i den kontext systemet ska användas i. Exempel på miljövariabler kan vara tillgång på kunniga medarbetare, vilken systemutvecklingsmetod som används, tid, samt kostnadsaspekter. Schwaber (1995) och Avison & Fitzgerald (2003) menar att dessa variabler förändras under projektets gång och det är det som ytterligare ökar komplexiteten och behovet av en form av metodik som klarar att anpassa sig till förändring. Detta innebär enligt Avison & Fitzgerald (2003) att man ska se systemutvecklingsprocessen som en odefinierad process.

Trots detta har det gjorts många försök att ta fram systemutvecklingsmetoder där man ser processen som definierad. Med definierad menas synen på att en process med ett visst antal givna inparametrar alltid producerar accepterbar kvalitet, i projekt efter projekt. Detta har i sin tur lett till att många organisationer varit oförberedda gentemot oförutsedda resultat och händelser (Larman 2004). Enligt Schwaber (1995) har detta resulterat i minskad tilltro till systemutvecklingsprojekt. En systemutvecklingsmodell som anser processen vara definierbar är den så kallade vattenfallsmodellen. Vattenfallsmodellen har legat som grund till ett stort antal systemutvecklingsmetoder och den började användas under 1970-talets början. Vattenfallsmodellen är linjär i sin karaktär, men är inte anpassad för att kunna hantera oförutsedda resultat i systemutvecklingsprocessen (Fitzgerald et al 2002). Detta trots att den har visst stöd för att hantera processen som en odefinierad process (Schwaber 1995).

Vattenfallsmodeller innebär i praktiken att man delar upp processen i tydligt definierade faser, och har man väl gått igenom en fas går man inte tillbaka till den igen. Detta i sin tur leder till att man ofta inte kan anpassa processen efter det som händer under projektets gång (Fitzgerald et al 2002). Exempel på möjliga förändringar hittar man i miljövariablerna, krav på ny funktionalitet etc. Även om många försök har gjort i att göra linjära metoder mer flexibla gentemot förändring, såsom iterativa varianter av vattenfallsmodeller, existerar fortfarande problemet att processen ses som definierad (Schwaber 1995).

Som ett svar på behovet av att kunna anpassa sig till förändring och synen på systemutvecklingsprocesser som en odefinierad process kom termen Agile upp i systemutvecklingsvärlden. 2001 samlades 17 framstående människor inom det agila fältet i Utah (USA) och samtalande om ämnet, resultatet av deras samlade erfarenheter blev Agile Manifesto, som är en programförklaring över de principer som stöttar agila systemutvecklingstekniker. Medverkande var bland annat Kent Beck, Ken Schwaber och Jeff Sutherland (Larman 2004). I Agile Software Development with Scrum (Schwaber 2002) förklaras Agile som en samling processer för systemutveckling, som använder iterativa och inkrementella tekniker samt förlitar sig på självorganiserande, självskötande och krossfunktionella arbetsgrupper.

Ett exempel på en Agile systemutvecklingsmetod med ett antal väl beprövade och tillförlitliga principer är XP (Extreme Programming), beskrivit enligt Kent Beck (2000, Preface XV) som:

*"A light-weight methodology for small to medium-sized teams developing software in the face of vague or rapidly-changing requirements."*

XP innehåller principer såsom parprogrammering, testning, kollektivt ägande av kod, kodstandarder, nära samarbete med beställare/kund etc. (Fitzgerald et al 2002). Agile innefattar således både processer och tekniker som man kan använda vid systemutveckling. För att hantera dessa skapade Ken Schwaber och Jeff Sutherland ett projekthanteringsramverk som de valde att kalla Scrum. Scrum har använts sedan tidigt 1990-tal och formaliserades 1995. Scrum definieras enligt Schwaber (2008, s1) som följande:

*"Scrum is an iterative, incremental process for developing any product or managing any work."*

Scrum som är en agil process ska inte ses som en systemutvecklingsmetod (Schwaber 2004), utan som en projekthanteringsprocess. I Scrumprocessen använder man sedan agila tekniker, som exempelvis kollektivt ägande av kod för att bedriva och stödja systemutvecklingsarbetet. Namnet är hämtat från rugby där Scrum är en tätt sammankopplad formation som tillsammans arbetar för att nå ett uppsatt mål.

Systemutvecklingsprocesser är enligt de agila förespråkarna mycket komplexa, vilket innebär att en systemutvecklingsprocess behöver empirisk processkontroll snarare än definierad (Larman 2004; Schwaber 2007). En empirisk process innebär att man använder sina tidigare erfarenheter för att guida sig genom processen, man lär över tid och utifrån det anpassar man processen allt eftersom den fortskrider (Schwaber 2007). I Scrum har man således inte en definierad process, man vet inte alla parametrar och resultatet av dessa. Scrum behandlar processen som en svart låda och förändrar den under iterationerna. Det kan handla om att lägga till eller ta bort tekniker som systemutvecklarna använder i sitt dagliga arbete (Larman 2004).

Synen på systemutvecklingsprocesser som empiriska innebär att man kan se Scrum som ett empiriskt angreppssätt och systemutvecklingsmetoder baserade på vattenfallsmodellen som ett definierat. Scrum har dock vissa definierade kontrollprocesser som används för att guida arbetet mot det mest värdefulla resultatet. Ett exempel på en kontrollprocess i Scrum är det dagliga mötet som kallas för Daily Scrum. Under detta korta möte stämmer systemutvecklargruppen av hur arbetet dagen innan gick. Scrum har således definierade

processer, framför allt i varje iterations planeringsfas, samt i slutfasen precis innan projektet avslutas. Andemeningen i Scrum är att det är processerna under systemutvecklingsarbetet som oftast är empiriska (Schwaber 1995).

Scrum innefattar tre roller, Product Owner, ScrumMaster samt Team. Product Owner som tar emot, hanterar förfrågningar om tillägg och eventuella ändringar i projektet. Det är också han som prioriterar vad som ska göras i en iteration utifrån största möjliga affärsnytta.

ScrumMaster kan sägas motsvara en projektledare och på hans ansvar ligger bland annat att synkronisera olika aktörer. En av hans huvuduppgifter är att se till att inget står i vägen för att teamet ska kunna genomföra sitt arbete. ScrumMaster ska hantera eventuella hinder som uppkommer för teamet, och lösa dessa. Det är ScrumMaster som ska utbilda deltagarna om Scrum, och det är även dennes ansvar att säkerställa att de definierade processerna efterföljs, som exempelvis att de varje dag har ett Daily Scrum. Själva teamet är de som utför systemutvecklingsarbetet och det ska vara en grupp på mellan fem och nio personer, de är självorganiserande och självbestämmande. Det är fritt för teamet att välja de processer de vill arbeta efter och med under en iteration, de kan exempelvis använda sig av tekniker hämtade från XP om de skulle finna dem lämpliga. Något som skiljer Scrum från mer traditionella projekthanteringsprocesser är att det är teamet själv som bestämmer mängden arbete de tror de kan genomföra under en iteration. Det läggs således en stor frihet hos systemutvecklargruppen, frihet och ansvar (Schwaber 2004). Genom denna frihet ska det skapas en mer kreativ, njutbar och produktiv arbetsmiljö, och i slutändan en produkt med maximal kvalitet och affärsnytta (Schwaber 2004).

Scrum är dock inget som passar alla organisationer och alla typer av projekt, det krävs exempelvis att beställaren av det som ska utvecklas engagerar sig och vill ta en aktiv roll i systemutvecklingsprojektet (Schwaber 2007). Scrum är ett enkelt och kortfattat ramverk, svårheten ligger i att anpassa Scrum till den aktuella organisationen. Om en organisation väljer att börja använda Scrum är problemet inte enbart att implementera Scrum, utan även att applicera ett agilt tänkande och medvetande hos utvecklarna. Något som ofta inte alltid är helt trivialt (Larman 2004).

Scrum lägger som nämnts ansvaret för att organisera arbetet på teamet, vilket kan resultera i svårigheter att anpassa sig till det nya självbestämmande och självorganiserande arbetssättet. Tidigare har kanske en projektledare bestämt vem i teamet som ska göra vad. Nu ska teamet själva sköta detta, utifrån vad det har bestämt är iterationens mål, och helt plötsligt ligger hela ansvaret på teamet. I Scrum är det inte en medlem av teamet som misslyckas, det är hela teamet som misslyckas och hålls ansvarigt (Schwaber 2004). Detta är bara ett par av de situationer som dyker upp när en organisation ska anamma Scrum och ett agilt tänkande.



## 1.2 Forskningsfråga

Vår forskningsfråga lyder;

*Vilken påverkan har en Scrum-implementering med XP-relaterade tekniker på en systemutvecklargrupps arbete, och vilka faktorer påverkar resultatet av implementeringen?*

Med arbete avser vi utförandet av de dagliga arbetsuppgifter kopplade till systemutveckling. Med påverkan en Scrum-implementering har på systemutvecklargruppen syftar vi på förändringarna i beteende, handlingar och åsikter, på individ och gruppnivå.

## 1.3 Syfte

Syftet med denna studie är att identifiera hur en systemutvecklargrupp påverkas av en Scrum-implementering, samt att identifiera de faktorer som påverkar resultaten av implementeringen.

## 1.4 Målgrupp

Vi har valt att utforma rapporten med tanke på medlemmar i organisationer som vill implementera Scrum i aktuell organisation. Medlemmarna kan vara systemutvecklare, projektledare, beslutsfattare etc. Den är vidare även utformad för forskarkollegor som vill öka sin förståelse för ämnet, därför beskriver vi Scrumprocessen och andemeningen med agil systemutveckling under kapitel tre.

## 1.5 Avgränsningar

För att anpassa omfånget av uppsatsen till de resurser och målsättningar man har är det en bra idé att tidigt lägga fram en rad avgränsningar som arbetet får rätta sig efter. Våra avgränsningar är följande:

Vi kommer att begränsa vårt sökande efter informanter till Sveriges gränser, med anledningen att våra personliga resurser inte tillåter oss att göra tillfredsställande intervjuer utomlands. Vi vill genomföra dessa intervjuer öga mot öga, istället för via e-post, telefon, eller liknande kommunikationsmedel. Anledningen till detta går vi djupare in på i metodkapitlet.

Vidare tänker vi avgränsa våra informanter till anställda inom privatägda teknikorienterade företag och grupper inom dessa som arbetar med mjukvaruutveckling. Denna avgränsning baserar vi på att dessa företag troligen är mer benägna att använda sig av en vattenfallsmodell i sin utveckling (Avison & Fitzgerald 2003). Schwaber (2007) menar att just en övergång från vattenfallsmodell till en lättroblig Scrum-implementering ökar

komplexiteten i implementeringen. Både lyckade och misslyckade implementeringar är av intresse för oss, då båda dessa utfall ger sina distinkta erfarenheter.

## 1.6 Disposition

Uppsatsen är uppdelad i fem huvudkapitel, bilagor och referenslista, samt ett speciellt kapitel med en expertintervju.

Kapitel ett utgör en introduktion till hela uppsatsen och går igenom bakgrunden till området. Det existerar problem inom dagens systemutveckling, och agila förhållningssätt (framförallt Scrum), kan vara en potentiell lösning till dessa. Vidare så hanteras uppsatsens problemområde, syfte, målgrupp och avgränsningar i detta kapitel.

Efter detta följer det kapitel som behandlar den teoretiska grund vi har använt oss av, och de olika delarna av vårt problemområde blir utsatta för separata genomgångar stödda av teorier funna i akademiska publikationer. Allt detta resulterar sedan i skapandet av en konceptuell modell över Scrums påverkan på utvecklingsgruppen, med entiteter och faktorer relevanta till just denna.

Kapitel tre är vårt metodkapitel, och innehåller en beskrivning över hur vi gått tillväga i skapandet av uppsatsen. Vår forskningsstrategi förklaras i detalj och följs av en genomgång av vår insamling och analys av empirisk data, även etik och forskningskvalitet tas upp här.

Det fjärde kapitlet är just denna analys, där vi går igenom vår empiri och sammanfattar den objektivt. Kapitel fem består av den speciella expertintervju som sedan bidrar till att belysa vissa frågor inför vår diskussion.

Sist kommer kapitel sex där vi med teori, empiri och analys genomför vår diskussion, och presenterar våra resultat och slutsatser, och således knyter ihop säcken.

## 2 Teoretisk bakgrund

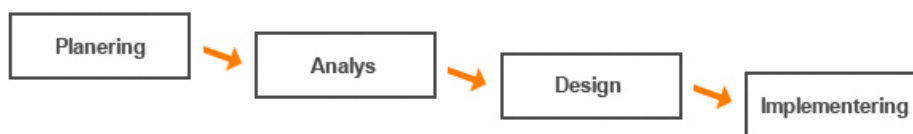
---

*I följande kapitel presenteras en teoretisk bakgrund över det som fallstudien ska undersöka. I slutet presenteras det som en konceptuell modell. Kapitlet behandlar Scrum, agilitet, traditionell systemutveckling etc.*

---

### 2.1 Agil systemutveckling kontra sekventiell

Under 1960-talet började systemutvecklingsforskningen hämta inspiration från processforskning och dess syn på livscyklar som ett sätt att arbeta gentemot problemlösning. Resultatet blev en livscykelmodell för systemutveckling där systemutvecklingsprocessen blev tydligt uppdelad i ett antal sekventiella faser. 1970 presenterades en livscykelmodell för systemutveckling där varje fas skulle vara avklarad innan nästkommande påbörjades. På grund av att fasernas ordning och faktumet att det var uppdelat efter en vattenfallseffekt kallades den vattenfallsmodellen (Avison & Fitzgerald 2003). Vattenfallsmodellen har haft en enorm påverkan på systemutveckling och blivit ett generellt förhållningssätt att utveckla informationssystem på (Avison & Fitzgerald 2003) och det finns ett mycket stort antal systemutvecklingsmetoder baserade på denna modell (Fitzgerald et al 2002; Larman 2004).



**Figur 2.1** Grunden för vattenfallsmodellen (Fitzgerald et al 2002)

Sekventiella systemutvecklingsmetoder baserade på vattenfallsmodellen ser systemutveckling som något som kan genomföras sekventiellt, en process som går igenom flera fördefinierade faser (se figur 2.1-1). Alla krav specificeras i en stor analysfas innan projektet sätter igång, sedan går man genom ett antal andra faser för att utveckla det man specificerat i analysfasen. Innebörden blir att det blir mycket svårt att anpassa utvecklingsprojektet när det under resans gång uppstår en ny kravbild på systemet (Szalvay 2004; Avison & Fitzgerald 2003). Avison och Fitzgerald (2003) menar även att detta kan orsaka missnöje gentemot den färdiga produkten hos slutanvändarna. Systemutvecklarna missar helt enkelt att anpassa produkten efter krav och åsikter hos slutanvändarna som uppstår under projektets gång. Styrkan med vattenfallsmodellen är att man får en trygghet i att ha en definierad process som man kan förlita sig på (Avison & Fitzgerald 2003), man vet vad man ska göra under processens gång och kan på det sättet använda diverse kontrollfunktioner för att säkerställa kvalitet (Fitzgerald et al 2002). Enligt Fitzgerald et al (2002) är ett av de största problemen med systemutvecklingsmetoder baserade på

vattenfallsmodellen kravet på att den tidigare fasen måste vara helt genomförd för att nästkommande ska kunna påbörjas, vilket kräver en perfekt bild av det som ska genomföras. Problem som kan uppstå i senare faser måste förutses i de tidigare faserna, vilket i realiteten ofta inte är möjligt. Fitzgerald et al (2002) menar att ett annat stort problem som vattenfallsmodellen bidrar med är antagandet om att alla krav till fullo kan bli förstådda i en initial planeringsfas. Detta ställer till problem då verkligheten oftast inte är så enkel, utan krav förändras under projektets gång. Ett annat problem med sekventiella modeller är att integration och tester oftast sker i avslutande faser, vilket kan leda till att problem uppdagas väldigt sent och resulterar i att projektet blir försenat, eller att det får en undermålig kvalitet. Agila systemutvecklingsmetoder som vi förklarar senare i detta kapitel för in testning under varje iteration. De levererar således en liten del av kvalitetssäkrad funktionalitet vid slutet av varje iteration (Larman 2004).

Fitzgerald et al (2002) poängterar att resultatet av de flesta faser är dokumentation, dokumentation som behövs för att veta vad man ska genomföra i nästkommande fas. Detta leder till att Fitzgerald et al (2002) definierar systemutvecklingsmetoder baserade på vattenfallsmodellen som tunga, med innebörden att de skapar mycket artefakter och är formella i sin karaktär. Även Avison och Fitzgerald (2003) menar att vattenfallsmodellen bidrar med problem kopplade till mängden och typen av dokumentation. Även om de anser en av fördelarna med en formaliserad systemutvecklingsmetod är en standardiserad typ av dokumentation finns det en risk med vattenfallsmodeller. Systemutvecklingsmetoder baserade på den skapar oftast en stor mängd teknikorienterad dokumentation vilket inte är idealt förenligt med syftet med dokumentation, nämligen kommunikation. De agila systemutvecklingsmetoderna däremot fokuserar främst på dokument som pratar kundens språk, för att tillsammans få en gemensambild av det som ska produceras. Teknikorienterad dokumentation blir ofta förlegad och skapar en mer tungrodd process (Larman 2004).

Systemutvecklingsmetoder baserade på vattenfallsmodellen fokuserar på grund av sin natur oftast mycket på hårda värden såsom teknik, den ser processen som definierbar och kontrollerbar (Fitzgerald et al 2002) vilket även kan leda till missnöje hos de som arbetar efter en sådan modell, de får för lite frihet (Avison & Fitzgerald 2003). Som ett alternativ till metoder med tyngd och fokusering på hårda värden framkom något som Fitzgerald et al (2002) kallar för de radikala förhållningssätten gentemot systemutveckling, förhållningssätt där man fokuserar på mjuka värden som människor och organisation, och framförallt på systemutvecklingsprojekts dynamiska natur. I dessa radikala förhållningssätt hittar man bland annat agila systemutvecklingsmetoder. Metoder som framförallt uppkom på grund av problem kopplade till bristen på flexibilitet gentemot förändring i vattenfallsmodellen (Fitzgerald et al 2002).

Agila systemutvecklingsmetoder applicerar tidsbegränsad iterativ och evolutionär utveckling. Tidsbegränsad i den innebörden att en iteration har ett kort tidsspann, ofta mellan två och fyra veckor. Med evolutionär menas att krav, planeringar, uppskattningar och lösningar utvecklas och omdefinieras över iterationerna. De fokuserar också på adaptiv utveckling vilket innebär att uppfattningar om sakers natur ändras allteftersom man får feedback på det arbete som genomförts (Larman 2004), tvärtemot systemutvecklingsmetoder baserade på vattenfallsmodellen (Avison & Fitzgerald 2003). Vidare menar Larman (2004) att man inte exakt kan definiera vad en agil systemutvecklingsmetod är, eftersom det finns en stor variation inom området. Dock innefattar de alltid värden och förfaranden som uppmuntrar agilitet, samt snabba och flexibla svar på förändring.

Abrahamsson et al (2002) fann i en studie över ett flertal agila systemutvecklingsmetoder följande gemensamma drag:

- Produkten utvecklas inkrementellt, små versioner av funktionell programvara släpps genom att arbeta i korta cykler.
- Kooperativt förhållningssätt, systemutvecklare och kunderna arbetar i ett nära samarbete.
- Rättfram och adaptiv, systemutvecklingsmetoden är lätt att lära sig och förändra, samt att den är snabbt kan svara på förändring av kravbilden.

De agila tankarna är inga nya idéer, tankarna har funnits inom systemutvecklingbranschen sedan 1960-talet, men det var först i början av 2000-talet de i större utsträckning började få uppmärksamhet (Larman 2004). 2001 träffades en grupp människor med auktoritet inom systemutvecklingsområdet och ett gemensamt intresse av iterativa och agila metoder för att skapa en gemensam grund för deras idéer. Resultatet av detta möte blev något kallat Agile Manifesto And Principles (se Bilaga 2) som innehåller fyra huvudpunkter och tretton principer som förklarar andemeningen med de agila systemutvecklingstankarna (Larman 2004), huvudpunkter är följande:

*We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:*

- ***Individuals and interactions over processes and tools***
- ***Working software over comprehensive documentation***
- ***Customer collaboration over contract negotiation***
- ***Responding to change over following a plan***

*That is, while there is value in the items on the right, we value the items on the left more. (Larman 2004, s.28)*

Viktigt att poängtera med dessa punkter är att man inte struntar i det som står på den högra sidan, men man lägger större betydelse på den vänstra sidan (Cockburn 2007). Något som Larman (2004) och Cockburn (2007) menar ofta feltolkas, att kritiker av manifestet tolkar det som att man exempelvis aldrig ska följa en plan. Kritiker menar att just detta lägger grund för så kallad Cowboy Coding och låg disciplin (Cockburn 2007), försvarare av manifestet menar att kritikerna således bara har feltolkat manifestet.

Första punkten behandlar synen på att programmering som del i ett systemutvecklingsprojekt är en mänsklig aktivitet och de agila principerna sätter individer och interaktion framför processer och verktyg. Dessa individer är den viktigaste punkten i en agil process (Larman 2004). I realiteten handlar det om att de agila systemutvecklingsmetoderna fokuserar i stor utsträckning på själva systemutvecklargruppen, dess dynamik och miljö (Abrahamsson et al 2002). Larman (2004) exemplifierar detta genom de agila metodernas tyngdpunkt på kommunikation som sker ansikte mot ansikte, kunskap förmedlas och deltagarna i gruppen lär sig genom att

interagera med varandra (Cockburn 2007). Det handlar om att se individerna i gruppen istället för att som i definierade processer, se individen som en specifik roll med en specifik uppgift (Cockburn 2007). En annan viktig princip kopplad till denna huvudpunkt är att anamma kommunikation och feedback, kommunikation genom att ha öppenhet och transparens i projektet. Transparens innebär att alla i projektet delaktiga personer kan överblicka projektets status. Feedback för att de agila metoderna är adaptiva och kräver feedback för att få möjlighet att stämma av processen, för att kunna justera den (Larman 2004).

Punkt två handlar om att de agila systemutvecklingsmetoderna fokuserar på att släppa versioner av produkten med utökad funktionalitet i slutet av varje iteration, detta genom återkommande testning av producerad kod under iterationens gång (Abrahamsson et al 2002). Programfunktionalitet får inte levereras eller anses som klar om inte den är noggrant testad (Larman 2004). Dokumentation är viktigt även i det agila manifestet men det ska användas i rätt utsträckning, dokumentation gällande krav, analys, design etcetera kan fungera som rådgivande stöd under processen men man ska inte förlita sig på det (Cockburn 2007).

Den tredje punkten innebär att man fokuserar på relationen och samarbetet mellan kunden och de som producerar systemet, detta för att det inte är troligt att kunden kan specificera allt produkten ska innefatta från start (Cockburn 2007). Det är något som tillsammans måste arbetas fram allt eftersom iterationerna fortskrider (Abrahamsson et al 2002). Ceschi et al (2005) påpekar att ett av de största problemen som agila systemutvecklingsmetoder adresserar är just problem relaterade till denna kundrelation. De menar att en högre grad av kundkontakt skapar en högkvalitativ länk mellan utvecklingsteamet och kunden, vilket gagnar för en bättre förståelse för det kunden i realiteten efterfrågar (Highsmith & Cockburn 2001; Fitzgerald et al 2002; Larman 2004). Dock poängteras det att detta är något som inte passar alla projekt och beställare (Cockburn 2007). I icke agila systemutvecklingsmetoder specificeras det ofta upp ett kontrakt som specificerar exakt vad som ska utvecklas, det går inte ihop med det agila tänket att man tillsammans arbetar fram funktionalitet efter största möjliga affärsnytta under iterationernas gång (Cockburn 2007). Kontrakt är viktigt för att definiera upp ansvar och åtaganden men ska inte vara detaljstyrande (Abrahamsson et al 2002). Cockburn (2007) menar att det är viktigt att ha en plan, och det är användbart att följa denna plan, så länge den inte är utdaterad. För att undvika en utdaterad plan har de agila metoderna förfaranden som skiljer dem åt, Scrum har exempelvis planering i starten av varje iteration (Cockburn 2007).

Punkt fyra innebär att de agila systemutvecklingsmetoderna måste kunna svara på uppkomna krav gällandes förändring. Larman (2004) anser att detta (Embrace change) är en av de viktigaste agila principerna. Att möjliggöra och acceptera att förändring är något oundvikligt som kommer att ske i alla projekt, man kan omöjligt veta allting när man startar ett projekt. Osäkerheten i ett systemutvecklingsprojekt är för hög för att man ska kunna bortse kraven på att vara anpassningsbar gentemot förändringar. Något som sekventiella modeller såsom vattenfallsmodellen ofta har svårt att hantera (Fitzgerald et al 2002).

De agila tankarna har fått utstå kritik, bland annat för att endast vara anpassade för mindre projekt med små systemutvecklingsgrupper, och det är delvis befogad kritik (Cockburn 2007). Agila metoder förutsätter transparens och kommunikation och möjligheten till detta försämras enligt Cockburn (2007) när man om man är färre än två eller mer än åtta personer i ett rum. Schwaber (2004) nämner mellan sju och nio personer i en grupp som det ultimata ur ett effektivitetsperspektiv. Men det finns enligt Cockburn (2007) inget som förhindrar att

en grupp på fyrtio systemutvecklare inte kan dra nytta av en agil systemutvecklingsmetod. De kan fortfarande vara agila men man måste anpassa hur agila de kan vara utifrån omständigheterna. Cockburn (2007) menar att den mesta kritiken som riktats mot agilitet överlag är baserade på missuppfattningar om vad agilitet egentligen innebär. Ett exempel han tar upp är påståendet att många agila projekt misslyckas på grund av att det är för lite disciplin, det blir ostrukturerat och ofokuserat, vilket enligt honom inte är felaktigt, men kritiken riktas fel då det agila fältet erkänner att agila metoder inte passar alla organisationer, då det kräver hög yrkesskicklighet och erfarna systemutvecklare med hög disciplin. Exempel på disciplin är att man inte kan göra antaganden om det som ska produceras, man måste säkerställa att man verkligen har förstått det så mycket att man kan börja producera (Cockburn 2007).

Cockburn (2007) tar upp det faktum att kritik har framlagts med påståendet att agila team inte behöver en plan, eftersom man i motsats till vattenfallsmodellen inte planerar hela projektet i starten av projektet. Detta för att principen i det agila manifestet (se bilaga 2) "Responding to change over following a plan" har misstolkats, manifestet menar att man ska värdesätta att följa en plan, men att man ska värdesätta att kunna svara på förändring mer (Cockburn 2007). Han menar att det i praktiken innebär att man ska göra en grundläggande plan, som under resans gång utvärderas. Man undersöker om planen verkligen hjälper till att nå målen, istället för att sätta upp en plan och gång på gång försöka komma tillbaka till denna plan när saker förändras.

Ett annat exempel som Larman (2004) och Cockburn (2007) tar upp är det Scrum och agila systemutvecklingsmetoder så som XP har utstått kritik för att vara emot dokumentation. Bland annat Stephens & Rosenberg (2003) menar detta. Larman (2004) och Cockburn (2007) menar att det också beror på att kritikerna inte har förstått innebörden av att vara agil. Agila systemutvecklingsmetoder och ramverk är inte emot dokumentation, men de är emot dokumentation som inte fyller någon funktion, för det skapar en tungrodd process med förlegad dokumentation. Larman (2004) menar att kritiken som riktas mot bland annat XP är befogad, men att det inte är något som XP sticker under stolen med. Ett exempel på kritik angående detta är att de flesta agila förhållningssätt kräver en aktiv kund, som verkar på samma arbetsplats som systemutvecklargruppen. Det är ett problem och befogad kritik eftersom detta inte är möjligt i alla projekt, men det är även ett krav i samt en del av XP, och många andra agila förhållningssätt (Larman 2004). Stephens & Rosenberg (2003) tar upp denna kritik men menar att XP har gått från att först kräva en kund på plats till att nu även föreslå ett team av affärsanalytiker som samarbetar med kunden. En variant av det Schwaber (2004) kallar för Proxy Product Owner.

Cockburn (2007) skriver om ett antal olika "Sweet Spots" för att optimalt kunna arbeta efter en agil systemutvecklingsmetod, och att om man har sin grund i en av dessa har man en god chans att nå ett lyckosamt utfall av sitt systemutvecklingsprojekt. Befinner man sig inte i en av dessa "Sweet Spots" ska man arbeta för att nå åtminstone en av dem menar han. "Sweet Spots" är enligt Cockburn (2007) följande.

- Två till åtta människor per arbetsrum – Information förflyttar sig snabbast och till minst kostnad under denna förutsättning.
- Domänexpert på plats – Med en person med stor kännedom om det som ska produceras får man möjlighet till korrekt feedback på frågor och antaganden. Detta ökar chansen för korrekta antaganden och en mer användbar slutprodukt.

- Inkrement på en månad – Inkrementell utveckling är perfekt för att få möjlighet till snabb feedback gällandes dels en förändrad kravbild men även på systemutvecklingsprocessen själv.
- Automatiska tester – Automatiska enhets- och funktionstest får utvecklarna att må bättre genom att de känner sig säkrare på att förändringar i programkod inte påverkar annan funktionalitet. Det ökar även kvaliteten på systemet i helhet.
- Erfarna utvecklare – Duktiga och erfarna utvecklare kan vara två till tio gånger effektivare än en mindre erfaren. Man kan använda ett färre antal utvecklare och få en bättre slutprodukt. Erfarna utvecklare kan förväntas ha den disciplin som agilitet kräver.

## 2.2 Agila tekniker och Extreme Programming (XP)

Eftersom Scrum är ett ramverk för att hantera projekt säger det inget om hur man bedriver det dagliga systemutvecklingsarbetet. Det är vanligt att Scrum används tillsammans med tekniker hämtade från systemutvecklingsmetoden Extreme Programming (XP) (Schwaber 2005). Här följer en kortfattad genomgång av några agila tekniker och tillämpningar (Practice) som bland annat återfinns i XP.

### 2.2.1 Unit Test och Test Driven Development

Ett Unit Test (se bilaga 17) är kod som testar en isolerad enhet av funktionalitet i kod, ett test kan antingen falla eller gå igenom. Testet skrivs av systemutvecklare och det är vanligt att man använder ett testramverk för att skriva sina tester såsom JUnit. Ramverket ger även utvecklarna möjlighet att köra automatiserade tester. Det vill säga att de enkelt kan köra alla tester som skrivits för det system som de utvecklar (Larman 2004).

Test Driven Development (TDD) är en programmeringsteknik som går ut på att man börjar utvecklingen av varje ny funktion med att skriva ett test (Unit Test), ett test som givetvis fallerar då man inte skrivit någon funktionalitet ännu. Man skriver sedan kod som endast uppfyller syftet för att implementera funktionaliteten, kod som får Unit Testet att gå igenom. Viktigt är att man endast skriver kod som är designad endast för att få testet att gå igenom, då man inte vill introducera ny otestad funktionalitet. Utvecklaren kör sedan de automatiserade testerna för att försäkra sig om att det han nyss implementerat inte orsakar att andra test fallerar. Efter detta rensar utvecklaren upp samt snyggar till koden, så kallad Refactoring. Även här har utvecklaren trygghet i att övrig kod också har tillhörande test (Beck 2002). TDD är en programmeringsteknik som kan användas i vilket systemutvecklingsprojekt som helst men är även en tillämpning i XP. Detta innebär att ett systemutvecklingsprojekt som använder XP innefattar tester av all funktionalitet (Larman 2004).

### 2.2.2 Coding Standards Coding Guidelines

Ett team som använder sig av XP har satt upp vissa riktlinjer för hur de ska skriva sin kod, så kallade Coding Standards eller Coding Guidelines. Detta är till för att koden ska vara enhetlig och se likadan ut oavsett vem i teamet som skrivit den. En av fördelarna är att



teammedlemmar enklare förstår delar av koden som de själva inte skrivit, de känner igen sig i den på grund av att den följer samma riktlinjer (Larman 2004).

### *2.2.3 Code Review*

Code Review innebär att utvecklare tillsammans går igenom producerad kod för att komma med förslag och tankar gällandes det som programmerats. Tankar och infallsvinklar som den som skrivit koden kanske inte tänkt på eller har haft. I XP sker Code Review i form av att XP kräver att man använder parprogrammering, vilket innebär att man alltid sitter tillsammans med en annan person när man skriver kod (Larman 2004). Syftet är att höja kvaliteten på det som programmerats.

### *2.2.4 User Stories och Use Cases*

En User Story (se bilaga 19) är ett kortfattat dokument (ett par meningar), vilket används som en del av specifikationen över kraven det som utvecklas har. Det används för att snabbt få en överblick över vad det är som ska utvecklas, utan att bli en tungrodd kravspecifikation. När en User Story ska implementeras skapas i XP ett mer formellt dokument (Acceptanstest) som gör det möjligt för kunden att senare kunna besluta om målet med User Storyn har uppnåtts (Cohn 2004).

Use Cases (se bilaga 18) används inte i XP men används ofta i andra typer av agila systemutvecklingsprojekt (Larman 2004). Ett Use Case är likt en User Story ett dokument som hanterar krav och dess funktionalitet. Ett Use Case innehåller ett eller flera scenario för varje fall systemet ska hantera, samt hur system integrerar med den som använder det.

Både Use Cases och User Storys är skrivna antingen av kunden eller i samarbete med kunden, med dennes språkbruk och begreppsvärld (Larman 2004).

### *2.2.5 Kollektivt ägande av kod*

Alla medlemmar i ett team som producerar kod har rätt att förbättra samt förändra varandras kod. För att detta ska kunna uppnås använder man sig av en gemensam lagringsplats (Repository) för den producerade koden (Larman 2004). Vidare menar Larman (2004) att det finns en fara i att ändra i kod som producerats av någon annan än en själv, men att denna fara ska försvinna med hjälp av andra tillämpningar inom XP, såsom Coding Guidelines, Unit Tests etcetera. Man får en trygghet i att känna igen sig i koden samt att man märker om ändringarna orsakat problem ifall Unit Testen fallerar (Cockburn 2007). En annan viktig poäng detta belyser är att ingen person hålls ansvarig för en viss del kod, utan teamet tillsammans är ansvarig för all producerad kod (Larman 2004).

### *2.2.6 Continuous Integration*

Continuous Integration är en viktig tillämpning inom XP (Fitzgerald et al 2002). Det innebär att man har en separat server som automatiskt bygger den senaste versionen av applikationen så fort ändringar läggs till i den gemensamma lagringsplatsen för kod. Det vill säga att så fort man lägger till kod till den gemensamma lagringsplatsen kompileras

koden och automatiserade tester körs. På detta sätt märker man om man har lagt till kod som orsakar problem i andra delar av applikationen (Larman 2004).

Schwaber (2004) menar att kollektivt ägande av kod och Continuous Integration krävs om man vill uppnå Scrums effektivitets fördelar, vilket förklaras i nästkommande delkapitel. I vattenfallsbaserade systemutvecklingsmetoder är det oftast förekommande att man istället kör tester och bygger applikationen i projektets slutfas (Larman 2004).

## 2.3 Scrum

Följande behandling av Scrum baseras till stora delar på material från källor som förespråkar Scrum på olika sätt, och kan därför verka partisk eller alltför positiv till Scrum. Vi har försökt att hålla oss neutrala till området, men det finns en risk att denna positiva retorik blir märkbar, då budskapet från källorna inte förändrats eller skrivits om. Läsaren bör vara medveten om detta.

Scrum definieras enligt en av upphovsmännen, Ken Schwaber (2008, s1) som följande.

*”Scrum is an iterative, incremental process for developing any product or managing any work.”*

Det innebär att det är en agil process eller ett ramverk för att hantera alla typer av agila projekt. Det ska alltså inte ses som en systemutvecklingsmetod utan som en projekthanteringsprocess, något som ofta feltolkas. Produktionsfördelarna med Scrum är att det ska resultera i en mer effektiv systemutvecklingsgrupp, och maximera affärsnyttan av det som utvecklas (Schwaber 2004).

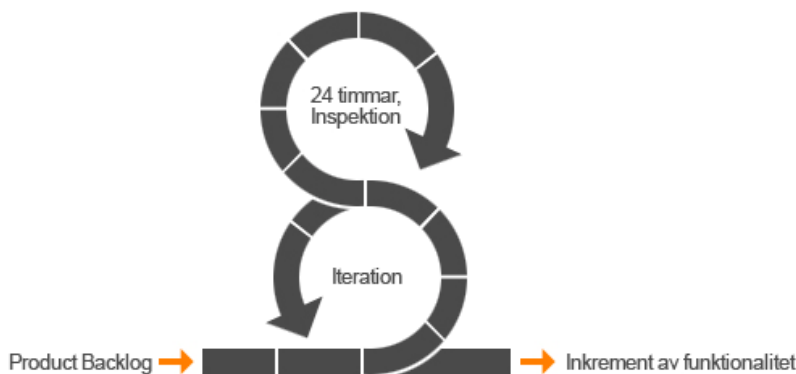
Scrum som term i produktutvecklingsprojekt uppkom 1986 i Japan där Ikujiro Nonaka och Hirotaka Takeuchi använde benämningen för en produktiv produktutvecklingsprocess. Namnet hämtades från sporten rugby där Scrum är en strategi där en sammansvetsad lagdel tillsammans arbetar mot ett specifikt mål (Schwaber & Beedle 2002). Scrum formaliserades av Ken Schwaber och Jeff Sutherland, och presenterades för första gången på OOPSLA (Object-Oriented Programming, Systems, Languages & Applications) 1996. 2001 skrev Ken Schwaber tillsammans med Mike Beedle boken Agile Software Development with SCRUM och det blev en sammanfattning av tidigare erfarenheter av Scrum. Namnet Scrum stannade kvar då de ansåg att rugbystrategin och den typ av produktutveckling de föreslog hade stora likheter. De är båda anpassningsbara, snabba och självorganiserande (Schwaber & Beedle 2002).

För att kontrollera en process finns det två frekvent använda modeller, den empiriska samt den definierade processkontrollen. Den definierade modellen kräver att man till fullo har förstått allt som ska genomföras. Har man en uppsättning väldefinierade indata, kommer resultatet bli det samma varje gång processen körs. En definierad process innebär således också att man kan starta den och köra den till dess att processen är helt genomförd (Schwaber & Beedle 2002). Det innebär att varje gång processen genomförs producerar den accepterbar kvalitet, i projekt efter projekt (Schwaber 1995). Systemutvecklingsmetoder baserade på vattenfallsmodellen använder sig av en definierad processkontroll (Larman 2004), men Schwaber (1995) menar att vattenfallsmodellen har visst stöd för att se processen som odefinierad men att det inte fungerar väl i praktiken.

En empirisk processmodell innebär en helt annan ansats till processer än den definierade, den är anpassad för att kunna hantera det oväntade, till skillnad från en definierad. Den empiriska tillhandahåller och genomför kontroll genom frekventa inspektioner och anpassningar av processen, då denna inte är fullt fördefinierad (Schwaber & Beedle 2002). I praktiken innebär det att man använder erfarenheter och lärdomar från tidigare skeden i processen för att definiera om den. Exempelvis anpassa processen efter de problem man stött på, processen ger en lärdomar och vidareutvecklas över tid (Schwaber 1995).

När Scrum togs fram samrådde upphovsmännen med en expertgrupp på områdena systemutvecklingsmetodik och industriell processkontroll. Babatunde Ogannaike som är en respekterad forskare inom industriell processkontroll ansåg att anledningen till att en stor del av alla påbörjade systemutvecklingsprojekt havererar är för att man försöker applicera en definierad processkontroll på en process som har för mycket osäkerhet och osäkra slutresultat. Det är utifrån detta Scrum tog sin empiriska form (Larman 2004). Schwaber (1995) och Larman (2004) menar att systemutvecklingsprojekt är komplexa och att de måste vara anpassningsbara för förändring, och kräver således en empirisk process med kontrollmekanismer. Scrum applicerar den empiriska processmodellens kontroll och inspektion genom att bland annat ha en arbetsmiljö som främjar kommunikation och öppenhet. Det skapar incitament för att kunna få en tydlig överblicksbild över vart arbetet är på väg (Schwaber 2004).

Scrum är en ganska kortfattad och lätthanterlig modell. Larman (2004) menar att Scrum är den minst detaljstyrda agila processen. Scrum innehåller tre roller, tre ceremonier och ett fåtal producerade artefakter (Schwaber 2004). Det bygger på ett iterativt, inkrementellt processskelett, som fungerar på följande sätt (se figur 2.3-1 Scrums processskelett).



**Figur 2.2** Scrums processskelett

Vid starten av en iteration, som i Scrum kallas för en sprint, undersöker utvecklargruppen vad de måste göra. De väljer där ut de uppgifter som de tror de kan omforma till ett inkrement av funktionalitet tills slutet av iterationen. En sprint sträcker sig vanligtvis över en 30-dagars period (Schwaber 2004). Utvecklargruppen får sedan efter sitt eget tycke göra det bästa möjliga för att nå slutmålet av iterationen. Det är just själva iterationen som är Scrums hjärta. Utvecklargruppen går igenom kraven, tar tillgänglig teknologi i betänkande, utvärderar sig egen förmåga och kapacitet. Sedan bestämmer de kollektivt hur de ska bygga utvald funktionalitet, samtidigt som de under hela iterationen utvärderar och förändrar sitt sätt att arbeta genom att de stöter på nya problem och hinder. Denna kreativa och stimulerande process är andemeningen i de incitament som ska skapa Scrums

produktivitet fördelar. Schwaber (2004) och Larman (2004) menar att just denna process är Scrum's viktigaste del.

### 2.3.1 Roller i Scrum

Processskelettet implementeras genom att det finns tre definierade roller i Scrum; ScrumMaster, Product Owner samt team.

ScrumMaster är den person som är ansvarig för att Scrum blir lyckosamt genomfört eller inte (Schwaber & Beedle 2002). Det är en roll som på flera sätt liknar en traditionell projektledarroll, med skillnaden att han eller hon inte detaljstyr det dagliga arbetet. ScrumMaster ansvarar för att teamet är fungerande och produktivt, ser till att roller och funktioner har ett nära samarbete och river barriärer inom dessa. En annan viktig huvuduppgift för ScrumMaster är att skydda teamet från externa störningar samt att säkerställa att processen efterföljs (Schwaber 2004).

Det är ScrumMaster som är ytterst ansvarig för själva Scrumprocessen. Det kan visa sig genom att han jämför vilka framsteg som skett i förhållande till vad man förväntade sig i sprinten, det vill säga att han övervakar processen. Han ska till exempel hålla ett vakande öga på teamet och se till att de har de förutsättningar som krävs för att de ska kunna arbeta så effektivt och stimulerande som möjligt. Det kan handla om att identifiera att en person har kört fast och har ett hinder som måste tas itu med. Det är även han som är ansvarig för att ta beslut som gäller teamets förutsättningar, finns det ett problem som hindrar teamet är det ScrumMaster som ska se till att lösa det, så länge det inte inkräktar på teamets självständighet och självbestämmande. Alla människor passar inte för rollen som ScrumMaster, personen måste inneha vissa specifika förmågor för att kunna riva eventuella hinder som står i vägen för processen, han måste vara initiativtagande och inte vara rädd för att synas. Det kräver också att han har beslutsamhet och är envis för att på ett effektivt sätt möjliggöra processens framgång (Schwaber & Beedle 2002). Schwaber (2007) menar att det inte är ovanligt att traditionella projektledare inte klarar av att hantera Scrum, de vill detaljstyra i en stor utsträckning. De är ofta vana vid traditionella systemutvecklingsprojekt och försöker ofta planera flera iterationer i förväg, något som går rakt emot innebörden av en iterativ process. Det är också ScrumMaster som ansvarar för att utbilda organisationen och de andra rollerna om Scrum, och se till att de till fullo förstår processen (Larman 2004). Kunskap om den aktuella processen som grund för lyckosamma systemutvecklingsprojekt är även något som Brittain White och Leifer (1986) fann i sin studie över systemutvecklingsprojekt.

Product Owner är ansvarig för att representera intressenterna i projektet och slutprodukten och måste vara en enda fysisk person (Schwaber & Beedle 2002). Schwaber (2004) tar upp att det är denna person som sätter upp de initiala kraven för det som ska utvecklas, vilken funktionalitet som ska uppnås, tar fram ekonomiska aspekter så som Return On Investment (ROI) och när produkten ska vara färdig. Hans uppgift är således att säkerställa att det som teamet utvecklar alltid görs utifrån ett maximalt affärsnyttoperspektiv. Ofta är Product Owner från en extern organisation, beställaren av ett system. Men denna individ kan även komma från den egna organisationen (Schwaber 2004). Produkt Owner väljs ut av ScrumMaster tillsammans med andra personer på en ledningsnivå (Schwaber & Beedle 2002), om projektet har en extern beställare är det givetvis beställare som i samråd med ScrumMaster väljer lämplig Product Owner. Det är en mycket aktiv roll och kräver en person med stort engagemang för det som ska utvecklas. På grund av att detta engagemang

måste fortskrida sprint efter sprint kan det uppstå problem ifall organisationen inte är van vid att arbeta på detta sätt. I traditionella systemutvecklingsprojekt är den som ansvarar för kraven kanske mest aktiv under den initiala planeringsfasen när specifikationen över vad som ska utvecklas tas fram (Larman 2004).

Teamet är själva utvecklargruppen, de som utvecklar systemets funktionalitet. Det är ett korsfunktionellt team, vilket innebär att det finns en blandad uppsättning kunskaper inom teamet, baserade på det som krävs för att nå sprintens uppsatta mål (Schwaber & Beedle 2002). Teamet ska bestå av sju, plus minus två medlemmar (Schwaber 2004). Teamet är själva ansvariga för att nå de mål som de sätter upp inför en sprint, inför varje sprint resonerar de fram hur mycket funktionalitet de kan producera under kommande sprint (Schwaber & Beedle 2002). Coplien och Harrison (2005) menar också att utvecklargruppen bör vara liten, fast de anser att 10 är en lämplig siffra. Att använda skickligare men färre antal människor är enligt Cohn och Ford (2003) centralt i en agil process. Detta förfarande behandlas mer i avsnittet om arbetsprocessen i Scrum. I traditionella systemutvecklingsprojekt är det oftast projektledaren som berättar för teamet vad som ska genomföras, och hur lång tid det kommer att ta, i Scrum läggs nu detta ansvar på teamet (Schwaber & Beedle 2002).

Schwaber och Beedle (2002) menar att produktiviteten minskar om antalet deltagare överstiger åtta, det leder även till att Scrums kontrollmekanismer blir svåra att hantera för ScrumMaster. De anser även att team med färre medlemmar än fyra kan dra nytta av styrkorna i Scrum men att det i sin tur kan vara kontraproduktivt då det begränsar mängden interaktion mellan deltagarna och på det viset minskar de produktivitetsförbättringarna som Scrum ska medföra.

Teamet är helt och hållet självorganiserande, den viktigaste aspekten av det är att de har all rätt att göra allt inom guidelinjerna för projektets ramar för att nå målen i en sprint (Schwaber 2004). Projektets ramar innebär att det ska passa in i förhållande till Scrum och organisationen som helhet (Schwaber & Beedle 2002). Under en sprint har teamet auktoritet att genomföra de beslut som krävs för att nå uppsatta mål. De har själva ansvaret att välja lämpliga tekniker, undersöka problematiker kopplade till uppgiften, söka råd av människor utanför gruppen etcetera. Om de sedan väljer att följa eventuella råd är upp till teamet själva.

En annan viktig aspekt med teamet i förhållande till organisationen är att de måste få existera i en miljö som främjar samarbete och produktivitet. De ska få tillgång till de bästa verktygen för att kunna utforma och utföra sitt arbete. Organisationen måste tillhandahålla utrustning och en arbetsmiljö som tillåter detta. En öppen arbetsmiljö främjar kommunikation och samarbete som ett exempel, det ska vara en arbetsmiljö som bara teamet verkar i, så att de inte blir påverkade av yttre störningar (Schwaber & Beedle 2002). Att ha en arbetsmiljö som främjar samarbete, kommunikation samt bidrar till att skydda gruppen för yttre störningar är även något som Larman (2004) poängterar är centralt för agil systemutveckling, dock finns det en organisatorisk fara menar han. Faran ligger i att om gruppen skärmar av sig från övriga organisationen och bryter mot befintliga normer och traditioner riskerar man att konflikter uppstår.

### 2.3.2 Ceremonier i Scrum

Scrum har tre ceremonier; Daily Scrum, Sprint Planning Meeting samt Sprint Review Meeting. Vi presenterar dem här väldigt kortfattat för att vidareutveckla dem i avsnittet Arbetsprocessen i Scrum.

Daily Scrum är ett dagligt möte på ungefär femton minuter som bland annat är till för att stämma av arbetet mellan medlemmarna i teamet, och de tillsammans med ScrumMaster är de enda som bör tala under detta möte (Schwaber & Beedle 2002).

Sprint Planning Meeting är ett möte där teamet, ScrumMaster och Product Owner träffas för att planera den kommande sprinten, även slutanvändare och andra för projektet viktiga personer kan medverka under detta möte. Efter mötet ska teamet vara på det klara med vad som ska genomföras under den kommande sprinten och ha en förståelse för hur de ska sätta igång med arbetet (Schwaber & Beedle 2002).

Sprint Review Meeting är ett möte som sker efter att en sprint har avslutats. Under detta möte presenterar teamet för Product Owner och andra intressenter vad de har åstadkommit under genomförd sprint. Mötets syfte är dels detta, men även att föra samman personer som har intresse i projektet, så att de tillsammans kan bestämma vad som är viktigt att producera under nästkommande sprint (Schwaber 2004). Teamet utvärderar även här den gångna iterationens arbetssätt, de diskuterar vad som var positivt och negativt med sättet de arbetade på, kan de förbättra något?

### 2.3.3 Artefakter i Scrum

Scrum har få artefakter, detta baseras på den agila synen att ett stort antal artefakter skapar en tungrodd process (Larman 2004).

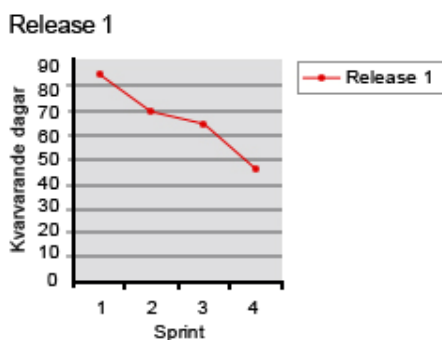
Product Backlog är en lista med funktionella krav och andra typer av krav angående den framtida produkten. Den innehåller således både affärsmässiga aspekter och teknisk funktionalitet som måste finnas med i produkten (Schwaber & Beedle 2002). En viktig aspekt är att punkter i denna lista inte direkt behöver kunna omsättas av teamet, utan kan fungera som påminnelser om saker som man måste utveckla i senare sprintar, det kan exempelvis vara krav som ännu inte är fullt specificerade, det är upp till Product Owner att förvandla dessa påminnelser till något som teamet kan sätta igång att arbeta med (Schwaber & Beedle 2002).

Product Backlog tas fram av Product Owner och kraven i listan prioriteras utifrån hans syn på vad som är värdefullt utifrån ett affärsperspektiv. Product Backlog innehåller även ett estimat på hur lång tid det tar att utveckla innehållet i listan. Detta estimat tas fram tillsammans med andra personer som har kunskap om teknik och utvecklingsprojekt, teamet är till stor del aktiva i framtagandet av denna uppskattning (Schwaber & Beedle 2002). Det är baserat på allt som är involverat i framtagandet av produkten, testning, arkitektur, konstruktion etc. Förmågan att göra korrekta estimat utvecklas i takt med att sprintarna fortskrider, och uppskattningen ändras allt eftersom de inblandande förstår mer om det som ska utvecklas. Viktigt att poängtera är att detta estimat inte är bindande för teamet, det är inte en specifikation över hur mycket tid de har på sig för att ta fram systemet, utan endast till för att få fram en kvalificerad gissning (Schwaber & Beedle 2002). Product Backlog är ett levande dokument som ändras under projektets gång, då förändringar i denna

representerar förändringar i kraven på systemet, det är dock bara Product Owner som får genomföra själva ändringarna i Product Backlog (Schwaber 2004). Men givetvis måste intressenter och eventuellt andra beställare av produkten kunna påverka Produkt Owner angående de krav som ska finnas med. För att Scrumprocessen initialt ska kunna starta måste denna artefakt vara producerad, åtminstone i så stor utsträckning så att den sammanlagda tiden för att producera kraven täcker den första sprinten (Schwaber 2004).

Sprint Backlog är den artefakt som innehåller detaljerad information om det som ska genomföras under aktuell sprint, punkterna har sitt ursprung i Product Backlog. Punkterna i listan ska vara utformade så att de tar mellan fyra och sexton timmar att genomföra, har de en större tidsåtgång ska man undersöka och definiera uppgiften tydligare och bryta upp det i flera mindre punkter (Schwaber 2004). Det är bara teamet som får genomföra ändringar i detta levande dokument. Det kan handla om att de vill bryta upp en punkt till flera mindre punkter i takt med att de får ökad förståelse av uppgiften (Schwaber 2007). Efter varje arbetsdag eller när en person har färdigställt en punkt uppdaterar personen Sprint Backlog. Eftersom man alltid anger hur många arbetstimmar det är kvar på varje enskild punkt får man på så vis en direkt överblick över läget i sprinten (Schwaber 2004).

En annan artefakt som Scrum kan producera är något som kallas för Burndown Chart. Om man tar ut en sådan gentemot Product Backlog får man fram ett diagram som visar mängden kvarvarande arbete över en tidsperiod (se figur 2.3-2). Det är ett sätt att visualisera sambanden mellan kvarvarande arbete vid en tidpunkt och projektets framsteg i färdigställda uppgifter. Diagrammet kan hjälpa till att visa vad som händer med projektets tidsåtgång om man lägger till eller tar bort uppgifter som ska utföras, dvs. funktionalitet. Man kan också få fram en Burndown Chart för en aktuell sprint, vilket ger teamet och ScrumMaster möjlighet att se sprinten status (Schwaber 2004). En Burndown Chart över sprinten är bland annat användbar för att se hur teamet ligger till i förhållande till kvarvarande tid och uppgifter, för att antingen plocka in eller plocka bort saker från sprintens backlog.

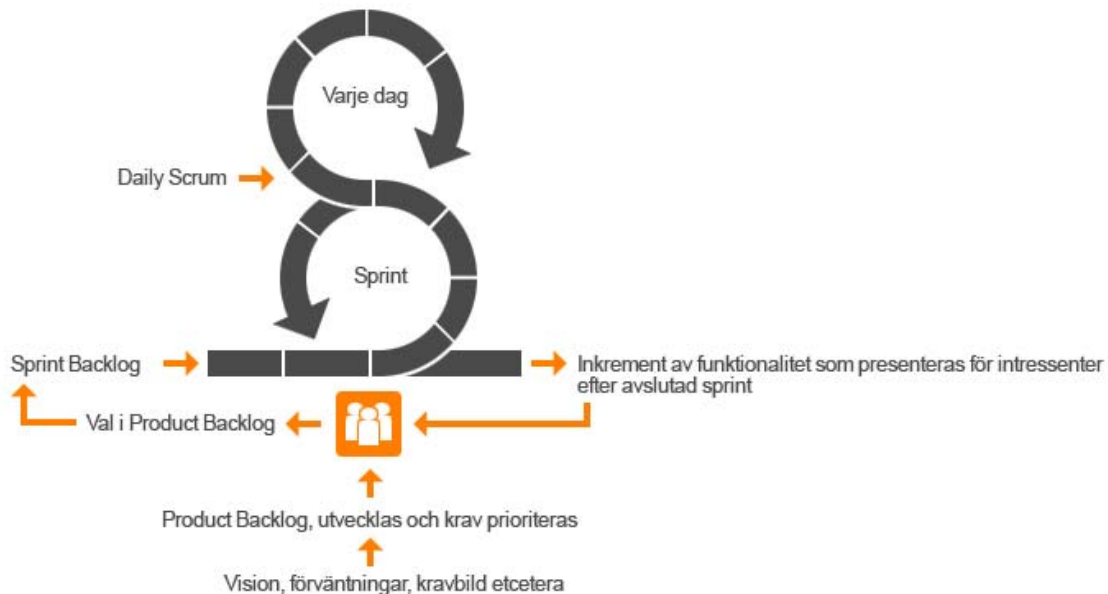


Figur 2.3 Burndown Chart

Scrum kräver att teamet producerar ett inkrement av produktfunktionalitet i varje sprint. Ett sådant inkrement ska bestå av testad, välskrivna och strukturerad kod, samt att användningen och testningen av denna kod är dokumenterad. Detta är definitionen av ett slutfört inkrement och det ska fungera i så hög grad att det skulle kunna sättas i produktion av Product Owner om så skulle behövas (Schwaber 2004).

### 2.3.4 Arbetsprocessen i Scrum

Arbetsprocessen (se figur 2.3-3, Scrum's arbetsprocess) i Scrum börjar med att Product Owner tar fram en Product Backlog och om möjligt ett initialt estimat på hur lång tid systemet kommer ta att utveckla. Estimaterna tas fram i samarbete med själva teamet. Som vi etablerat tidigare bedrivs allt arbete i iterationer som kallas för sprintar. En sprint är vanligtvis mellan två och fyra veckor lång (Schwaber & Beedle 2002).



Figur 2.4 Scrum's arbetsprocess

Varje sprint börjar med att man genomför ett planeringsmöte för sprinten, det så kallade Sprint Planning Meeting, där Product Owner tillsammans med Scrumteamet går igenom samt diskuterar fram vad som ska göras i den aktuella sprinten. Product Owner tar ut de krav från Product Backlog som har högst prioritet och lägger fram vad han önskar få ut av sprinten. De i sin tur går igenom hur mycket av kraven de kan omvandla till funktionalitet under den kommande sprinten. Detta möte får maximalt ta 8 timmar, annars riskerar man att hamna i för mycket diskussion om vad som kan genomföras, poängen är att komma igång med arbetet och inte diskutera arbetet in absurdum (Schwaber 2004). Schwaber och Beedle (2002) menar att i de första sprintarna finns det en risk för att teamets uppskattning av vad de kan genomföra under sprinten inte är optimal. Denna risk kan uppstå om Product Owner inte har tagit fram estimat för hur lång tid punkter i Product Backlog tar tillsammans med teamet. Men då det är en empirisk process ligger ansvaret för att korrigera detta på ScrumMaster, så att det i efterföljande sprintar blir ett mer korrekt estimat.

Mötet är indelat i två delar, under den första halvan presenterar Product Owner de krav han vill få med från Product Backlog. Teamet ställer då frågor om innehållet i kraven, deras syfte och mening för att få en ökad förståelse för det som ska tas fram. Detta är en kreativ process där det verkligen är upp till teamet att säkerställa att de har förstått Product Owner intentioner och begreppsvärld (Schwaber 2004). Innan första halvan är slut och när teamet vet tillräckligt om kraven från Product Backlog, väljer de ut det som de tror att de kan genomföra. Nu formuleras ett övergripande mål för sprinten, baserat på de punkter som har valts ut från Product Backlog (Schwaber & Beedle 2002). Under andra halvan av mötet



planerar teamet hur de ska bedriva arbetet i sprinten för att nå det övergripande målet. Eftersom de är självbestämmande när det gäller hur de utför sitt arbete behöver de en plan för att kunna sätta igång arbetet med sprinten. De punkter och uppgifter som ska genomföras för att nå målet förs in i Sprint Backlog, som likt Product Backlog är ett levande dokument och modifieras under den kommande sprintens gång (Schwaber & Beedle 2002).

Ceremonin med Daily Scrum har nämnts tidigare, och har som syfte att synkronisera arbetet hos teammedlemmarna och planera in eventuella möten som behövs för att komma vidare med eventuella problem som uppstått. Mötet startas genom att ScrumMaster frågar varje teammedlem följande tre frågor (Schwaber & Beedle 2002; Schwaber 2004; Rising & Janoff 2000).

- Vad har du gjort sedan föregående möte?
- Vad tänker du göra inför kommande möte?
- Är det något som hindrar dig från att uträtta ditt planerade arbete?

De första två frågorna ger deltagarna full insikt hur projektet går, den tredje är till för att säkerställa att eventuella problem kommer upp till ytan, det kan vara allt från tekniska till organisatoriska förändringar som påverkar aktuell sprint. Detta möte är också ett sätt att vänja de inblandande personerna vid kommunikation, öppenhet och samarbete (Schwaber & Beedle 2002).

Sprint Review Meeting, som nämnts tidigare, genomförs när en sprint avslutas, och är ett fyra timmars informellt möte där teamet presenterar resultatet för intressenter och Product Owner. De visar upp den funktionalitet de utvecklat. Mötet är till för att intressenter och Product Owner gemensamt ska kunna diskutera vad som ska göras närmast, hade de tänkt rätt inför denna sprint, finns det andra saker som kommit upp till ytan nu som måste bejakas inför nästa Sprint Planning Meeting? (Schwaber 2004)

Under en sprint får inga ändringar i form av nya krav på vad som ska utvecklas under sprinten läggas till. Om Product Owner upplever att det måste läggas till saker i sprinten finns två alternativ. Det första är att Product Owner avbryter sprinten, och man börjar om med planeringsmötet. Det andra alternativet är att Product Owner går till ScrumMaster och frågar honom om han i sin tur kan kolla med teamet om de kan ändra i sprinten. Teamet får då avgöra om de klarar av att ta med det, och vad som måste plockas bort för att få in Product Owners önskemål.

Innan nästkommande Sprint Planning Meeting genomför ScrumMaster ett möte med teamet där föregående sprint diskuteras, Sprint Retrospective Meeting. Utifrån Scrumprocessen försöker Scrum Master lyfta fram saker som teamet kan göra bättre i nästa sprint, och tillsammans utvärderar de alltså sitt arbete. De tre mötena under en sprint är hjärtat av Scrum's empiriska process. Man gör en empirisk inspektion av den aktuella Scrum-processen och hur teamet använder tekniker kopplade till denna process. Man lär sig således med tiden. Det är meningen att man efter varje sprint ska förfina sin process, för att på så sätt nå en mer tillfredsställande arbetssituation och producera en bättre produkt (Schwaber 2004).

## 2.4 Problematik kopplat till att arbeta agilt

När man använder Scrum löser man problem och frågor under systemutvecklingsprocessen tillsammans, alla förväntas engagera sig och bidra till lagarbetet. Schwaber (2007) menar att det inte är ovanligt att människor inte vill arbeta på detta sätt, de vill hellre ha någon som säger åt dem vad de ska göra när de är på arbetsplatsen. Han menar att detta innebär att hela gruppen blir lidande och personen i fråga borde bytas ut inför nästkommande sprint.

Schwaber (2007) skriver om att utvecklare har ett sorts muskelminne skapat av hur de tidigare har arbetat, och att detta muskelminne är opassande och negativt för Scrum. Han menar att när det uppstår problem i Scrumprocessen så som stress, ett stort svårslösligt problem, tenderar utvecklare att gå tillbaka till sätt att arbeta som är kontraproduktivt för Scrum. Det finns fyra sorters muskelminne som förhindrar Scrums förmåga att förändra systemutvecklingsprocessen; Vattenfallstänk (Waterfall Thinking), beordra och kontrollera (Command And Control), åtagande till att trotsa naturlagarna (Commitment to Defying the Laws of Nature) samt att dölja verkligheten (Hiding Reality).

Enligt Schwaber (2007) uppkom vattenfallsmodellen ur projektledares vision om att man kan överbygga komplexitet med sannolikhet, och det är en modell som haft stor påverkan på hur vi utvecklar informationssystem. Vattenfallsmodellen har varit den dominerande utvecklingsmodellen sedan mitten av 60-talet och den används fortfarande i stor utsträckning (Cockburn 2007). Scrum skiljer sig som tidigare nämnts avsevärt från detta sekventiella tänk. I vattenfallsmetoder bygger man inte funktionalitet innan krav, arkitektur och infrastruktur till fullo är specificerade och dokumenterade menar Schwaber (2007). Schwaber (2007) exemplifierar detta med att det är vanligt för nyblivna användare av Scrum att försöka specificera upp flera sprintar i förväg och definiera hur arbetet kommer att utveckla sig i förväg. I Scrum arbetar teamet gemensamt med de olika delarna i projektet, testning, programmering, dokumentation etcetera, en traditionell vattenfallsanvändare ser uppgifterna som anpassade för en specialist på den aktuella uppgiften.

Att beordra och kontrollera handlar om synen att det är systemutvecklargruppen som själva bäst beslutar om hur de ska utforma sitt arbete, inte deras ledning. Schwaber (2007) menar att om man är bunden till en utomstående persons instruktioner är man inte självständig och fri nog för att utföra sitt arbete på det effektivaste sättet. Han påpekar att det är svårt för projektledare att bli fria från att vilja beordra och kontrollera arbetet och det är vanligt att de faller in i detta kontraproduktiva förfarande när det uppstår problem. Om teamet märker att det är för mycket uppgifter i Product Backlog för aktuell sprint och således frågar ScrumMaster om hur de ska göra, då ska inte han lösa det åt dem. Han ska guida dem genom processen över att dekonstruera Product Backlog, så att teamet nästa gång vet hur de självständigt ska lösa problemet. Varje gång ScrumMaster säger åt teamet vad de ska göra utövar han beordring och kontroll, han måste istället anta uppgiften att vara ansvarig för att lära teamet självstyre och problemlösning. Enligt Schwaber (2007) är det absolut största hindret för Scrum arvet från vattenfallsmetodiken, där personer har mer definierade roller. Denna avsaknad av rolltillhörighet kan få användare att gå tillbaka till detta arv, människor vill ofta ha en specifik roll för att känna sig bekväma i sin identitet och arbetsroll. När det gäller att få ett team att arbeta korsfunktionellt som ett team istället för en grupp av individer som arbetar sekventiellt menar Schwaber (2007) att det bara finns ett svar, ett svar som kan vara svårt för många att acceptera. Man ska inte göra någonting utan vänta på att gruppen själv ska organisera sig och bryta sina gamla mönster, vilket sker naturligt då de initiala sprintarna inte brukar vara så effektiva om man inte arbetar som ett korsfunktionellt

team. Denna självorganisering är givetvis svårare än det traditionella sättet utifrån en projektledares perspektiv, det kan inte skyndas på vilket normalt en projektledare kan gå in och delvis styra. ScrumMaster kan ge insikt och guidning till teamet i denna fråga, men ett korsfunktionellt sätt att arbeta kan inte skapas genom att tvingas fram.

Åtagande till att trotsa naturlagarna handlar enligt Schwaber (2007) att anta en utmaning som är ogenomförbar. Det handlar om att Product Owner kan lägga fram att ett antal uppgifter måste vara med i den kommande sprinten, ett antal uppgifter som tidsmässigt är ogenomförbara. Ett åtagande kan antingen vara falskt eller omöjligt att genomföra, ett åtagande är något som bygger på tillit och tilltro, skulle man inte klara av ett åtagande finns det risk för att skada denna tillit. Enligt Schwaber (2007) så bidrar traditionellt vattenfallstänk negativt till falska och omöjliga åtaganden, enligt honom är det troligare att utvecklare som utnyttjar vattenfallstänkande antar för stora åtaganden. Detta på grund av att de projekten ofta drar ut på tiden och detta i sin tur sätter sig i systemutvecklarnas muskelminne.

Att dölja verkligheten handlar om problem kopplade till att Scrum är en empirisk och transparent process. Transparent innebär att Product Owner och teamet alltid ska kunna se den egentliga statusen på projektet. Scrum handlar i slutändan om att få fram en produkt med så hög kvalitet som möjligt vilket inte är möjligt utan denna transparens. Problemen kan exemplifiera sig i att teamet inte hinner klart med alla uppgifter i en sprint, de avstår från kvalitetssäkrande processer så som testning och kodgenomgång för att hinna klart. De visar upp uppgifterna som genomförda för Product Owner under det kommande Sprint Review mötet, när de i själva verket inte är genomförda. Detta gör de inte av illvilja utan för att de vill visa upp att de klarat av sina åtaganden, och att de tror att de kan gå tillbaka och genomföra det som de struntat i under nästkommande sprint. Detta förfarande går helt emot Scrum då endast fullt genomförda uppgifter ska visas upp för Product Owner, de måste även berätta för honom att i nästa sprint ska de göra saker som de redan sagt de gjort, vilket de naturligtvis inte kan göra, för då erkänner de att de inte hunnit klart (Schwaber 2007).

De flesta utvecklare möter införandet av agila metoder med en kombination av skepticism, entusiasm och försiktig optimism. Men vissa motarbetar försöket till förändring, eller hoppar överentusiasmerat in i projektet, båda förloppen kan skapa problem (Cohn & Ford 2003). Cohn och Ford (2003) studerade en organisation som gick över till XP där en del av utvecklargruppen var överentusiasmerad, vilket resulterade i stora problem och orsakade splittring och konflikt inom gruppen. Den överentusiasmerade gruppen började planera och utveckla efter XP-tekniker fast de inte hade den disciplin som krävdes för det. De tog felaktiga beslut i sin iver över att få använda en agil metod, vilket skapade en grupp som tyckte att de gick för snabbt fram och att de gjorde för många felaktiga antaganden. De hade tolkat XP rekommendationen "put in what you need when you need it" som om de bara behövde planera en timme i förväg, vilket resulterade i en mindre katastrof. De fann också att en del utvecklare inte ville anpassa sig till förändringen som en agil process medför, de var vana vid en formell metod och ansåg att de agila var för kaotiska. De gick och klagade hos personalavdelningen över att de inte trivdes med arbetsmiljön och de ville tillbaka till att arbeta sekventiellt, vilket i sin tur skapade oro i organisationen.

Cohn och Ford (2003) studerade vad som händer när man introducerar en agil process i en organisation. De menar att man i icke agila processer producerar ett stort antal artefakter som inte har med programkod att göra, till exempel UML-diagram. Agila metoder fokuserar främst på artefakter som har med programkod att göra, och när de inte har det är

de oftast till för att stödja framtagandet av dessa, såsom User Stories och Use Cases. När de studerade införandet av Scrum fann de att många utvecklare gillade att ta fram artefakter som inte var till för att stödja framtagandet av programkod, såsom diverse generiska dokumentmallar. I agila processer ska man försöka undvika dokument som inte är till för att stödja framtagandet av programkod. Dock används UML i agil systemutveckling, det är bara hur man använder det som är det viktiga (Larman 2004). Mycket dokumentation och formella procedurer lägger grunden för en tungrodd process (Schwaber & Beedle 2002).

Coplien och Harrison (2005) menar att systemutvecklare har en stark tradition av att programmera och designa system efter sitt eget tycke och smak. Detta kan gå emot den agila synen att ett system ska vara utformat så att alla snabbt kan sätta sig i det och ska vara utvecklat efter en enhetlig kodstandard. Att motarbeta denna individualism och försöka bryta gamla mönster är en uppgift som måste genomföras, och den ökar vi-känslan inom gruppen enligt författarna. Det gemensamma ägandet av kod kan enligt Coplien och Harrison (2005) ställa till problem inom ett team. Systemutvecklare vill ofta äga och vara ansvariga för sin egen kod, samt att det tar tid att sätta sig in i andra människors kod om man inte följer en kodstandard.

Coplien och Harrison (2005) är också av åsikten att ett tätt samarbete mellan kund och systemutvecklare är ett måste, något som även är en agil huvudprincip (Larman 2004). Utvecklarna måste ha möjlighet till kommunikation med kund, och det måste finnas en tradition inom företaget för detta. En formell process som vattenfall har negativ inverkan på detta beteende enligt Coplien och Harrison (2005), men det är samtidigt inte alltid möjligt att få denna tillgång till denna kommunikation menar de.

## **2.5 Psykologi, grupper och gruppdynamik**

West et al (2001) definierar en grupp som en social entitet inom ett företag vars individuella komponenter (gruppmedlemmarna) arbetar för att slutföra arbetsuppgifter. För att åstadkomma detta måste de interagera med varandra och samarbeta, samt lägga sitt förtroende i varandras händer, då de olika deluppgifterna inom gruppen är sammankopplade. Detta leder till att en individs arbete kan hindras ifall någon annan gruppmedlem inte fullföljt sin personliga arbetsuppgift (West et al 2001; Cohen & Bailey 1997).

Fördelar med arbete inom grupper är sådana som att utveckling inom en organisation kan genomföras snabbare och mer effektivt, att kunskap och information kan hushållas på ett mer optimerat vis, och att kvalitetshandling kan förbättras (West et al 2001). En av faktorerna som bidrar till allt detta är förmågan att ha korsfunktionella grupper, där ett brett spektrum av erfarenheter och kunskaper ryms, och tack vare teamets utformning i Scrum som just en sådan grupp försöker man i Scrum ta del av dessa fördelar (Natale et al 1995; West et al 2001; Schwaber & Beedle 2002).

Samtidigt existerar det en rad nackdelar, som till exempel att gruppmedlemmar anpassar sig alltför mycket till varandra, vilket resulterar i mindre vågade beslut, "Groupthink" (då individer är mer bekymrade över att tillfredsställa de övriga medlemmarna än att göra ett bra arbete) samt ett utspänt ansvarstagande (West et al 2001; Witte & Davis 1996). Med det sistnämnda menas att alla felaktigt tror att andra kommer att axla ansvaret för besluten och resultaten av arbetet, vilket leder till att ingen till slut känner sig ansvarig (West et al 2001).

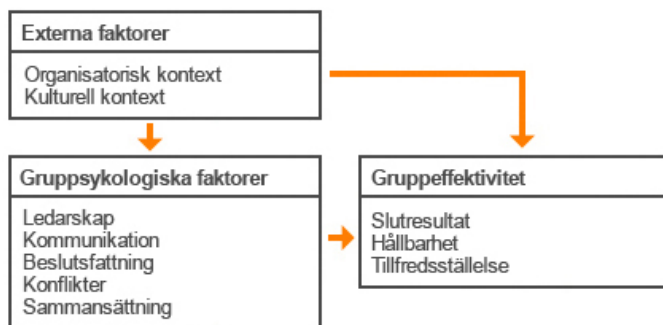
Detta ser även Schwaber och Beedle (2002) som ett problem, då en huvudpunkt med transformationen till att bli en självbestämmande grupp är att den måste lära sig att kollektivt ta ansvar. Ansvaret ligger aldrig på individen själv, det är gruppen som helhet som är ansvarig för projektet.

### 2.5.1 Grupp effektivitet

Vad gäller effektiviteten hos en grupp så finns det flera olika teoretiska ramverk och modeller, där en av de mer koncisa är den som beskrivs både av West et al (2001) och Cohen och Bailey (1997), nämligen grupp effektivitet bestående av tre övergripande faktorer:

- Slutresultatet, i form av kvalitet och kvantitet på prestationerna som gruppen producerar.
- Hållbarheten på gruppen, som innefattar stärkt samarbetsförmåga, engagemang, samt höjt förtroende för arbetsprocesserna inom organisationen.
- Tillfredsställelse hos de anställda, nämligen att deras behov tillgodoses, samt deras säkerhet och hälsa.

Dessa faktorer påverkas i sin tur av en rad olika källor, som visas i följande modell, baserad på texterna av West et al (2001) och Cohen och Bailey (1997):



**Figur 2.5** Modell över grupp effektivitet och dess påverkande faktorer. Punktlister innehåller exempel, och är inte uttömmande.

Det som går att utläsa ur modellen är alltså att grupp effektiviteten påverkas av externa faktorer samt de interna grupp psykologiska faktorerna. Det finns även en direkt koppling mellan de två förstnämnda genom att de interna ändrar karaktär beroende på de externa.

Enligt Coplien och Harrison (2005) är effektiviteten i ett nybildat team låg i starten av ett projekt. De menar att det beror på att gruppen måste vänja sig vid att arbeta tillsammans och organisera sig som grupp. Men det kan även bero på att gruppen saknar en gemensam vision av det som ska genomföras. Teamets nära samarbete med Product Owner och att det ska finnas tydliga mål för hur en sprint används i Scrum för att överbygga dessa hinder (Schwaber 2004). Att ha ett uniformt mål inom gruppen hjälper till att bygga lagandan, och på så sätt få en bättre grund för självorganisation (Coplien och Harrison 2005).

Ett annat problem kopplat till självorganisation och effektivitetsförbättringarna det innebär, är att teamet behöver reell erfarenhet av Scrum innan de kan inse till full grad hur de ska

organisera sig och hur de ska ta ansvar, samt auktoritet för att planera och utföra sina aktiviteter. Schwaber (2004) menar att ett av de vanligaste problemen i denna transformation är att teamet måste inse att Scrum kräver komplett transparens. Annars kan medlemmar i teamet ta fel beslut på grund av att de har en felaktig bild av hur arbetet har fortskridit.

Cohn och Ford (2003) fann i studier om införandet av agila processer att det initialt sker en effektivitetsförsämring när man går över till en agil process, det tar tid att lära sig nya tekniker och det är viktigt att organisationen är medveten om detta, så att det inte uppstår panik. Tanken är ju att det ska bli en effektivitetsförbättring, problemet är att den i många fall inte är omedelbar.

## 2.6 Gruppsykologiska faktorer och dess inverkan på effektivitet

Det existerar ett antal faktorer inom grupparbete som påverkar effektivitet, och i följande delkapitel kommer dessa att tas upp. De första två som vi kommer att ta upp, ledarskap och kommunikation, utgör samma problemområde, nämligen koordination. Medlemmarnas arbete måste vara noga koordinerat, för att arbetet inte ska bli oorganiserat och oproduktivt (West et al 2001). Övriga faktorer som vi kommer att behandla är beslutsfattning, konflikter samt sammansättning.

### 2.6.1 Ledarskap

West et al (2001) nämner tre ansvarsområden för en gruppleddare: Att hantera gruppen (genom att följa framstegen och fördela roller), att leda gruppen (för att åstadkomma långsiktiga mål) samt att ledsaga enskilda gruppmedlemmar genom att hantera problem på en lägre nivå. Här stöter vi på en markant skillnad mellan traditionella projekthanteringsmetoder och Scrum, då ScrumMaster inte är tänkt att arbeta på så låg detaljnivå inom gruppen (Schwaber 2004). Då det istället är teamets eget ansvar att organisera sig är det upp till dem att utse inre ledarskap.

### 2.6.2 Kommunikation

Coplien och Harrison (2005, s.311) skriver:

*“Communication is a complex human activity filled with social context, psychological complexity, and emotion. So, building a communication structure that in turn supports the building of a product is also a complex activity that requires a cultural setting conducive to effective communication.”*

Exempel på viktiga faktorer när det gäller kommunikation inom en grupp är att denna kommunikation sker på ett sätt som alla inom gruppen är medvetna om och har anammat, att utbytet av information och kunskap är ett givande sådant, samt att medlemmarna faktiskt är motiverade till att kommunicera med varandra (West et al 2001; Law & Charron 2005; Judy & Krumins-Beens 2008). Ett problem kan vara att det helt enkelt är för mycket byråkrati kring kommunikationen. Inom Scrum är det (som tidigare nämnts) tänkt att Daily Scrum Meeting inte enbart ska agera som ett öppet och lättillgängligt forum för denna kommunikation, utan även främja och utveckla teamets förmåga och vilja att genomföra

den på ett konstruktivt sätt (Schwaber & Beedle 2002; Schwaber 2004). Detta synsätt på kommunikation är dessutom en av de viktigaste principerna i agila metoder (Larman 2004). Daily Scrum Meeting är till för att öka projektets transparens, något som Coplien och Harrison (2005) anser vara av betydelse då förändringar i projektet måste nå ut till alla medlemmar i gruppen.

En viktig skiljelinje att dra enligt West et al (2001) är den mellan kommunikation som sker i samband med ett gruppmöte, och kommunikation som berör enskilda gruppmedlemmar. När det gäller den senare så förespråkar Law och Charron (2005) en arbetsmiljö där gruppmedlemmarna arbetar i ett gemensamt område, istället för att befinna sig utspridda i en byggnad eller organisation. Detta ger understöd till kommunikation i den mån att mindre frågor blir enklare att ställa, samt att de byråkratiska hindren försvinner när det gäller kunskapsutbyte. Då Scrum inte dikterar något speciellt förfarande för kommunikation utanför dess tre ceremonier, är denna del upp till teamet själva (Schwaber & Beedle 2002; Schwaber 2004). Schwaber (2004) anser att detta är något som kommer att ske naturligt. Detta på grund av att om de inte har öppenhet och effektiv kommunikation kommer de troligtvis misslyckas med de första sprintarna, och själva organisera om sig för att förhindra att det händer igen.

### 2.6.3 Beslutsfattande

Beslutsfattande är en central faktor när det gäller gruppeffektivitet, och påverkas i sin tur av ett par fenomen. Det viktigaste är vem som egentligen utför besluten, och huruvida övriga gruppmedlemmar har något att säga till om. Beslut kan liknas vid problem som behöver lösas, och ifall det finns en gruppkultur att lösa problem tillsammans kan detta leda till en ökad kvalitet i besluten som gruppen producerar (West et al 2001).

Karakowsky et al (2004) instämmer i detta, och kombinerar det med korsfunktionalitet, som synes i följande stycke (2004, s.506-507):

*“Numerous researchers have found that the recognition of expertise in a team is important for effective decision making [...]. A commonly-cited advantage of team decision making over individual decision making is the multitude of perspectives, information and expertise that members bring with them to addressing team problems [...]. However, a team composed of members who possess the relevant task expertise will nonetheless fail to maximize the team’s potential decision-making effectiveness if these members fail to exert influence in the team”*

Scrums ramverk över teamet och dess inre processer (transparens, kontrollmekanismer etc) passar perfekt för ett gemensamt beslutsfattande. Det finns ingen i förväg utsedd ledare som styr, och ifall gruppen arbetar fram en inre ledare så är det fortfarande på gruppens villkor, eftersom det är de som tillsammans ska se till att arbetsmiljön blir produktiv (Schwaber & Beedle 2002; Schwaber 2004).

Schwaber (2004) anser transformationen från att vara ett team som leds av en projektledare till att bli ett självständigt och självhanterande team är mödosam och svår. Men att det är något som måste genomföras då den resulterande produktiviteten och arbetsmiljöförbättringen är stor. Transformationen är något som är extra problematiskt om teamet tidigare har arbetat med traditionella systemutvecklingsmetoder, då de i dessa troligtvis har blivit tillsagda exakt vad de ska göra och när. Schwaber (2007) menar att

projektledare vill styra teamet på detaljnivå och personen som agerar ScrumMaster måste därför våga lämna över de besluten till teamet. Cohn och Ford (2003) instämmer i detta och tillägger att de (teamet) initialt kan vara överrumplade av den frihet de har att planera sitt arbete, då det inte finns någon projektledare som beordrar dem.

Studier av Cohn och Ford (2003) visar på att team som använder en agil process tenderar att ta beslut snabbare än team som inte använder en agil process. Detta på grund av att agila processer förlitar sig på en hög grad av kommunikation och transparens, något som Scrum kräver. Moore et al (2007) håller med och tillför att dåliga snabba beslut kan vara till större nytta än ett långdraget beslut, då man mycket snabbare kan upptäcka brister och alternativa lösningar till problemen.

#### *2.6.4 Konflikter*

Konflikter går att kategorisera i två grupper: uppgiftsrelaterade och relationsrelaterade (De Dreu & Van Vianen 2001). Även om det kan verka vara en enkel indelning av konflikttyper passar den in på vårt problemområde, vi ville ej heller bli för detaljerade angående olika konflikttyper. Av dessa är de senare betydligt svårare att hantera, på grund av att känslorna hos de inblandade är starkare. När det gäller en uppgiftsrelaterad konflikt är det vanligare att gruppen går samman och med enad kraft löser problemet. Ytterligare en skillnad är att vid en relationskonflikt så är det lättare för de inblandade att hålla en låg profil med sina åsikter, vilket gör att problemet kvarstår och påverkar effektiviteten indirekt (De Dreu & Van Vianen 2001; Peslak 2005). Trots detta kommer De Dreu och Van Vianen (2001) fram till att relationskonflikter påverkar effektiviteten mest negativt när de tas upp och hanteras, med motiveringen att de då drar fokus ifrån arbetet som ska göras. Detsamma gäller inte arbetsuppgiftskonflikter, då dessa ofta måste lösas för att arbetet inom gruppen ska kunna fortskrida. Det bör även tilläggas att konflikter även kan ha positiva konsekvenser, så som ökad individuell kreativitet, ökad kommunikation och bättre slutligt beslutsfattande (De Dreu et al 1999).

Higgs et al (2005) tar upp den intressanta poängen att heterogena grupper har en större tendens att hamna i konflikter, på grund av deras vitt skilda medlemmar. Detta är något som bör bejakas vid sammansättningen av en grupp då denna ska arbeta agilt (Larman 2004).

#### *2.6.5 Sammansättning*

Sammansättningen av en grupp är ytterst viktig för att denna ska kunna producera bra resultat, och teorierna som berör detta område är uppdelade i två läger. Å ena sidan har vi de forskningsresultat som påvisar att en heterogen grupp är mest fördelaktig, och på andra sidan har vi de som menar att heterogenitet är bra, men att grupper behöver sammansättas av ett bredare spektrum av medlemmar (West et al 2001; Wilkinson & Fung 2002; Mello & Ruckes 2006).

Återigen återknyts detta till faktumet att i Scrum är teamet tänkt att vara korsfunktionellt. Själva tanken bakom detta är att det aldrig ska saknas någon speciell expertis eller färdighet (Schwaber & Beedle 2002; Schwaber 2004). Detta gör skapandet av ett team till en viktig del av Scrumprocessen, något som vi återkommer till senare.



## 2.7 Externa faktorer

Det är dock inte bara de interna gruppsykologiska faktorerna som är avgörande, utan även externa faktorer. West et al (2001) förklarar att forskningen inom detta område inte har nått samma utsträckning som grupprelaterad forskning, och att detta beror på svårigheterna kring att jämföra olika kontexter. Det som behövs är ett antal olika organisationer, som alla har liknande grupper som arbetar på sätt som påminner om varandras (West et al 2001). Det finns dock en del att säga om detta område ändå.

### 2.7.1 Organisatorisk kontext

West et al (2001) beskriver ett par olika synsätt man kan använda för att förklara den organisatoriska inverkan på grupparbete. Den enklare av dessa bygger på tre övergripande punkter: belöningsystem, information och feedback, samt träning. En mer intressant lista erbjuds dock av Tannenbaum et al (1992 via West et al 2001), och består av följande:

- Belöningsystem (individuella eller gruppbaseade)
- Resursbrist
- Ledningskontroll
- Nivå av stress inom organisationen
- Organisationsklimat
- Konkurrens
- Grupprelationer inom organisationen
- Osäkerheter i organisationsmiljön

Intressantast av dessa i samband med Scrum är resursbrist, ledningskontroll, grupprelationer och osäkerheter. Som Schwaber och Beedle (2002) skriver så förutsätter Scrum att teamet får tillgång till de nödvändiga resurserna, och här faller ansvaret för detta på både ScrumMaster och Product Owner. De ska se till att teamet har organisationens stöd i form av både resurser och entusiasm för projektet. ScrumMaster är även den som blir utsatt för de övriga tre intressanta faktorerna, då dessa utgör just den form av yttre påfrestningar som teamet ska besparas (Schwaber & Beedle 2002; Schwaber 2004). Detta ansvar ligger även delvis på teamet, då det enligt Schwaber och Beedle (2002) ofta är förekommande att individer utanför projektet ber medlemmar i teamet om hjälp med för projektet irrelevanta problem. Det ligger i människans natur att vilja vara till lags men gruppmedlemmarna måste enligt författarna fokusera på sin arbetsuppgift, och vara restriktiva vad gäller de flesta yttre önskemål och krav som sträcker sig utanför projektets ramar.

### 2.7.2 Kulturell kontext

Även kulturell kontext har sin plats i diskussionen kring gruppeffektivitet, och hur man ska uppnå en hög sådan. West et al (2001, s.164) nämner vissa intressanta skillnader:

*”The social loafing effect identified in Western societies [...], is apparently non-existent and sometimes reversed in China and Israel [...].”*

Samt även West et al (2001, s.164):

*”The effects of participation, in particular, are noteworthy. Current Western thinking is that group participation is useful in enhancing performance [...]. However, this is based upon studies performed in individualistic, low power distance cultures. Cultures with a high power distance, on the other hand, have reported negative effects of participation [...].”*

## 2.8 Att sätta samman en grupp

Schwaber (2007) skriver att det är Product Owner och ScrumMaster som är ansvariga för att välja vilka människor som ska ingå i teamet. För att optimera produktiviteten ska de väljas utifrån tre variabler, människor som tidigare och lyckosamt har arbetat tillsammans, människor som förstår produkt- eller affärskontexten samt människor som vet hur man ska använda vald teknologi. Inför en sprint kan teamet omformas för att passa målet med sprinten, men detta ska endast göras när det är absolut nödvändigt. Det orsakar en produktivetsförsämring då teamet måste omorganisera sig och på nytt normalisera sig. Schwaber (2007) menar dock att det inte är lika omstörtande att plocka bort en medlem ur gruppen som när man för in en ny person.

Schwaber (2007) poängterar att man kan välja ut ett par personer som passar in i projektet för att sedan låta dem välja ut resterande medlemmar. Det är en teknik som även Coplien och Harrison (2005) förespråkar. Det är troligare att människor med samma värdegrund arbetar effektivare tillsammans än personer med vitt skilda värdegrunder. Higgs et al (2005) menar att antalet konflikter riskerar att öka ju mer heterogen en grupp är, vilket ytterligare ger tyngd åt Coplien och Harrisons (2005) påstående. Om en grupp blir sammansatt utan att man tänkt på hur de fungerar tillsammans finns risk att man skadar gruppdynamiken, och på det viset få en negativ inverkan på produktiviteten (Coplien & Harrison 2005).

Coplien och Harrison (2005) menar att interpersonliga förhållanden i en grupp har en stor negativ eller positiv påverkan av effektiviteten i ett team. De menar att det måste finnas en stor tillit mellan människorna inblandade i ett projekt om de ska kunna arbeta effektivt, kommunikation är betydelsefull för denna tillit. Systemutvecklare i ett team måste kommunicera med varandra för att kunna koordinera arbetet.

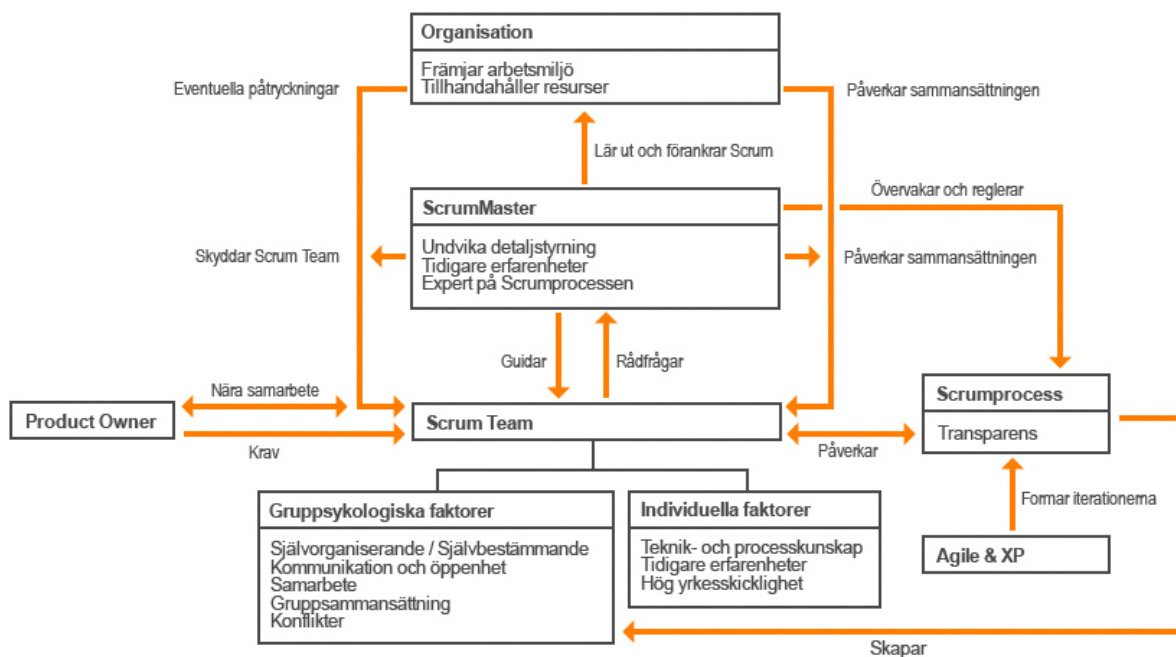
Daily Scrum och transparensen i processen är enligt Schwaber (2007) ett sätt att öka denna tillit, att teamet är självorganiserande en annan faktor. De måste kommunicera och ha en stor öppenhet för att lyckas med de uppsatta målen. Människor i ledningsnivå måste även de förmedla tilltro till teamet, ScrumMaster måste då öppet visa att han ansvarar för att hjälpa teamet och inte kontrollerar eller styr deras utvecklingsprocess. Att ScrumMaster visar detta öppet bidrar enligt Coplien och Harrison (2005) till att tilliten till varandra ökar hos teamet.

Coplien och Harrison (2005) menar att det även är viktigt att bygga och premiera teamets kollektiva styrka samtidigt som man uppskattar och tar tillvara den individuella styrkan hos var och en av medlemmarna. Om man har en övertro på en individs styrka och sätter honom som ensam specialist på ett område, finns risk för att man underminerar gruppen och de andra medlemmarnas identiteter och självförtroende. Att kollektivt arbeta gentemot en utvecklingsuppgift innebär att man har flera som bidrar till lösningen, då motiverar man individerna samtidigt som man genomför en kreativ kunskapsspridning. En människa med

specialistkunskaper sätter troligtvis riktningen i problemlösning inom sitt område, men den kollektiva diskussionen genererar incitament för en bättre lösning (Coplien & Harrison 2005). Reel (1999) är av samma uppfattning när det gäller individer med specialistkunskaper, dessa kan fungera som ledsagare och visa vägen för andra gruppmedlemmar. Han menar att denna specialistkunskap är nödvändig för ett lyckosamt projekt, alla kan inte vara lika bra på alla saker.

## 2.9 Konceptuell modell

Resultatet av vår genomgång i detta kapitel gav oss möjlighet att skapa en konceptuell modell (se figur 2.9-1) över hur införandet av Scrum kan påverka en systemutvecklargrupp. Det är de faktorer som påverkar gruppen som vi har valt att belysa i denna modell.



Figur 2.6 Konceptuell modell

En förklaring av modellen följer:

### 2.9.1 Faktorerna som påverkar Scrum Team

En rad olika faktorer påverkar det dagliga arbetet för teamet. Dessa kan i sin tur kategoriseras som antingen individuella faktorer eller gruppykologiska faktorer.

De individuella faktorerna inkluderar fenomen som kunskap om tekniken och processen, tidigare erfarenheter, samt yrkesskicklighet. Dessa har en direkt påverkan på medlemmarna i teamet då de är en central del av det dagliga individuella arbetet. Yrkesskickligheten krävs för att teamet ska kunna arbeta självständigt och kunna ta egna beslut, mindre erfarna systemutvecklare tenderar att förlita sig på andra för att driva arbetet framåt, på grund av osäkerhet etcetera (Schwaber 2004). Yrkesskickligheten innefattar således även stor kunskap om det som ska utvecklas, så kallad domänkunskap. Cockburn (2007) menar även

att en förutsättning för att lyckas med agilitet är att använda duktiga och erfarna utvecklare, då systemutvecklargruppen har relativt få medlemmar krävs det att alla kan bidra. Tidigare erfarenheter är något som kan bidra positivt och negativt. Positiv påverkan handlar om att en utvecklare för med sig kunskap som han kan använda för att öka teamets effektivitet, det kan handla om att han är expert på något område, kunskap som han sedan sprider inom gruppen. Negativ påverkan kan innefatta det som Schwaber (2007) kallar för muskelminne hos systemutvecklare, som exempelvis att man för in ett sekventiellt vattenfallstänk i en grupp som ska arbeta iterativt och inkrementellt, vilket leder till att man motverkar effektivitetsförbättringarna det agila tänkandet ska medföra. Om en individ saknar kunskap om Scrumprocessen, kommer inte personen att känna trygghet i sättet att arbeta. På det viset missgynnas effektivitetsfördelar såsom transparens och självorganisering som Scrum ska leda till, och på detta vis haltar Scrumprocessen som helhet (Schwaber 2007). Detta i sin tur kan leda till att ScrumMaster försöker gå in och detaljstyra teamets arbete, Schwaber (2007) poängterar att detta kan hända då mindre erfarna ScrumMasters ska guida team som har problem med att anpassa sig till självorganisation och självbestämmande.

De grupprelaterade faktorerna, med exempel som samarbete, kommunikation och öppenhet, självorganisation och självbestämmande är skapade av Scrumprocessen. Som ett exempel ökar tilliten mellan medlemmar i gruppen på grund av transparensen Daily Scrum medför, samt den öppna arbetsmiljön som teamet ska verka i (Schwaber 2004). Scrum förutsätter och förlitar sig på dessa faktorer för att nå de effektivitetsfördelar som dessa faktorer bidrar med, fördelar som ökad produktivitet, bättre arbetsmiljö, bättre slutprodukt etcetera. Självorganisation och självbestämmande är det som Schwaber (2007) och Cockburn (2007) poängterar är mest mödosamt att nå, det är ofta en kostsam process att nå dit. Detta på grund av att det är teamet själva som måste göra denna resa och upptäcka fördelarna detta bidrar med, det brukar ta ett par sprintar att nå dit (Schwaber 2004). Detta leder också till att de första sprintarna inte brukar vara speciellt produktiva, gruppens energi går åt till att transformera sig till att bli självbestämmande och självorganiserande, man gör misstag som man sedan lär sig av och förändrar sitt arbetssätt (Cockburn 2007). Scrums empiriska del träder således in (Schwaber 2004). I och med denna mödosamma resa riskerar man också att ScrumMaster försöker kontrollera och beordra teamet, vilket går helt emot dennes uppgift och arbetsroll (Schwaber 2007). Det kan handla om att teamet själva måste lära sig att lösa konflikter inom gruppen, att själva lösa de problem som uppstår i det dagliga och inte förlita sig på att någon annan, som exempelvis ScrumMaster ska lösa dessa problem åt dem. När det gäller sammansättningen av gruppen bör den enligt Schwaber (2007) bestå av personer som delar samma värdegrund och grundläggande syn på saker och ting. Detta för att det är troligare att det resulterar i mindre konflikter och en mer stimulerande arbetsmiljö, man jobbar med människor som delar ens värderingar, något som också stöds av Harrisons (2005). Gruppsammansättningen formas givetvis även av Scrumprocessens rekommendationer angående storlek och krav på korsfunktionalitet. Har inte gruppen rätt sammansättning riskerar den att inte kunna nå de uppsatta målen och bidra negativt till gruppens transformation till en självorganiserande enhet (Schwaber 2004).

### *2.9.2 Organisationens eventuella påtryckningar på Scrum Team*

Som nämnt tidigare, så utövar organisationen i en traditionell arbetsmiljö påtryckningar på arbetsgruppen, i form av orelaterade arbetsuppgifter, ändrade tidsscheman och arbetsuppgifter, samt liknande störande inslag. Ansvaret faller då på ScrumMaster att motverka dessa påtryckningar och därmed tillåta teamet att arbeta vidare enligt deras egen arbetsplan.

Ytterligare en aspekt till detta förhållande är att ScrumMaster måste arbeta för att lära ut och förankra Scrum i organisationen, så att alla involverade är insatta i varför och vad som måste genomföras för att lyckas med Scrum. Förstår individer inte varför, kan individer komma att motarbeta Scrumprocessen och på det viset försena eller förhindra effektivitetsfördelarna som ska uppnås. Det blir helt enkelt en ond cirkel som kan vara svår att komma ur (Schwaber & Beedle 2002). Detta inkluderar även att organisationen i form av avdelningschefer etcetera ska se till att teamet får tillgång till de resurser de behöver för att kunna genomföra projektet så effektivt som möjligt. Teamets sammansättning måste vara så att den kommer klara av projektets uppgifter, teamet ska vara korsfunktionellt.

Organisationen måste även tillhandahålla arbetsmiljö som främjar kommunikation och öppenhet, detta för att få möjlighet att nå Scrums effektivitetsfördelar som dess transparens medför (Schwaber & Beedle 2002). Cockburn (2007) menar att det krävs en öppen arbetsmiljö där teamet sitter nära varandra och att de då får ta del av något han kallar för brus (Noise), vilket innebär att även om man inte är med i en pågående diskussion hör man ändå delar av den och på det viset ökar transparensen inom gruppen. Alla organisationer vill inte låta systemutvecklargrupper arbeta i en miljö som beskrivits ovan och det är då upp till ScrumMaster att försöka få fram en lösning på detta problem, det är som tidigare nämnts han som ska lära ut Scrumprocessen inom organisationen.

### *2.9.3 Organisationen och ScrumMasters påverkan på gruppssammansättningen*

Själva sammansättningen av gruppen är som tidigare nämnts väldigt viktig för att teamet ska komma att lyckas med självorganisation och självbestämmande (Schwaber & Beedle 2002). Gruppen ska vara korsfunktionell, det är en relativt liten systemutvecklargrupp som måste klara av ett brett spektra av uppgifter (Schwaber 2004).

Som vi nämnt tidigare påpekar Schwaber (2007) samt Coplien och Harrison (2005) att ett bra sätt att sätta samman ett team är att välja ut ett par personer som ska vara medlemmar av gruppen och sedan låta dessa välja ut övriga medlemmar. De är troligare att de väljer personer som de vet kommer fungera i grupp, än någon högre upp i organisationen. Cockburn (2007) poängterar dock att detta förhållningsätt till gruppssammansättning inte är så lätt att driva igenom, det brukar finnas motstånd inom organisationer mot det. Men han menar även att det ökar chanserna för en lyckad sammansättning. Alltså ökar chansen för en bra sammansatt grupp om organisationen främjar detta sätt att arbeta med gruppssammansättning. ScrumMaster påverkar också gruppens sammansättning då denne ofta är delaktig i att välja ut vilka gruppmedlemmar som krävs för att lyckas genomföra ett projekt, det är även han som ska se till att utöka gruppen med lämpliga deltagare om det skulle uppstå problem kopplade till att gruppen saknar någon form av expertis (Schwaber 2007).

### *2.9.4 Product Owner och dennes koppling till teamet*

Product Owner är nära kopplad till teamet, då han eller hon är ansvarig för utvecklandet av krav för arbetet som ska genomföras, samt skapandet av artefakter så som Product Backlog.

Samarbetet sker i formen av att Product Owner tillsammans med teamet under Sprint Planning Meeting diskuterar vad som ska göras och planerar den kommande sprinten. De krav som Product Owner arbetat fram tas upp till diskussion, där medlemmarna i teamet

kan komma med frågor, förslag, och förtydliganden (Schwaber 2004). Förstår inte teamet Product Owners önskemål och bild av det som ska genomföras kommer inte den kommande sprintens resultat bli tillfredställande. Det är då upp till teamet själva att förbättra sitt arbete gentemot Product Owner inför nästa sprint. Schwaber och Beedle (2002) menar att systemutvecklare ofta är ovana att arbeta så nära kunden som Scrum kräver, och det är något som kan påverka resultatet av de första sprintarna. Tills det att teamet själva lärt sig hur de ska arbeta gentemot Product Owner för att nå de uppsatta målen, ligger det i teamets eget intresse att ha ett bra och givande samarbete med Product Owner (Schwaber 2004).

Det finns också problematik kopplat till att Product Owner är så delaktig i Scrumprocessen jämfört med hans delaktighet i exempelvis en sekventiell systemutvecklingsmetod där han har den största delaktigheten i de initiala faserna. Känner teamet att Product Owner inte ger dem det stöd de behöver är det upp till ScrumMaster att se detta hinder och undanröja det, att få Product Owner att inse nyttan av det nära samarbetet och höga delaktigheten (Schwaber 2004). Återigen ligger det på ScrumMasters roll att förankra och lära ut Scrum. Det är också väldigt viktigt att teamet är ärliga och öppna gentemot Product Owner, och inte döljer åsikter och tankar kring det som produceras. Teamet är experter och har varit med om systemutvecklingsprojekt förut och kanske ser problem som inte Product Owner har förutspått (Schwaber & Beedle 2002). Att tillsammans arbeta fram en gemensam vision för arbetet och slutresultatet ser Judy & Krumins-Beens (2008) som en viktig faktor.

#### *2.9.5 ScrumMasters roll i Scrumprocessen, och kopplingen till Scrum Team*

Då Scrum inte fungerar som en vanlig systemutvecklingsmetod med en utnämnd projektledare som bestämmer hur arbetet ska gå till, är förhållandet mellan ScrumMaster och teamet lite annorlunda.

Även om ScrumMaster inte styr detaljer, så kan medlemmar i teamet be honom eller henne om råd, till exempel vid problem i arbetet. ScrumMaster ska även övervaka gruppen och hela processen, och vara vaksam för potentiella problem som gruppen står framför, det kan exempelvis handla om att de är i behov av en extern expert för att kunna komma vidare med arbetet med målen för aktuell sprint (Schwaber 2007). Det är på detta sätt som ScrumMaster guidar teamet, och de i sin tur rådfrågar ScrumMaster (Schwaber 2004). Försöker ScrumMaster detaljstyra teamet kommer han att bidra till att förhindra gruppens transformation till att bli självstyrande och självbestämmande, och på det viset skapa en negativ påverkan på hela projektet (Schwaber 2004). Detta är inte ovanligt om ScrumMaster har lång erfarenhet av traditionell projektstyrning, men det är även något som kan ske naturligt då stressade situationer uppstår menar Schwaber (2007). Han poängterar dock att det är något som både ScrumMaster och teamet måste vara vaksamma på, men ytterst ligger ansvaret på ScrumMaster, han måste låta teamet styra sig själva (Schwaber 2007).

Detta förhållande gäller både kopplingen mellan ScrumMaster och teamet, samt kopplingen mellan ScrumMaster och Scrumprocessen. Detta på grund att ett av ScrumMasters ansvarsområden är att övervaka och reglera Scrumprocessen. Han reglerar den genom att tillsätta resurser, undanröja hinder, vara den som kallar till och håller i möten såsom Daily Scrum etcetera. Han övervakar den genom att vaksamt lyssna på teamet under dessa möten, han övervakar hur arbetet fortskrider genom att undersöka sprintens status via Burndown Charts (Schwaber 2004).

### *2.9.6 Scrumprocessens arv från det Agila tänkandet*

Scrumprocessen ärver mycket av de tankegångar som återfinns i det agila manifestet. Detta påverkar sedan teamet direkt via faktorer som ökad kommunikation, transparens, etc. Individerna i processen ska dessutom sättas före verktygen. När en systemutvecklargrupp arbetar efter Scrum använder de agila tekniker för att utföra sitt dagliga systemutvecklingsarbete, det kan vara att de använder sig av tekniker såsom Code Review och Pair Programming (Schwaber 2004). Continuous Integration och ett kollektivt ägande av kod är något som krävs för att Scrum ska kunna få så hög transparens som möjligt (Larman 2007).

### *2.9.7 Teamet och Scrumprocessen*

Teamet påverkas av Scrumprocessen eftersom de måste anpassa sig till arbetssättet som är förknippat med Scrum. De måste följa de procedurer som är till för att skapa transparens i projektet, exempelvis måste de vara noggranna med att fylla i Sprint Backlog, närvara och ärligt berätta om aktuell status på Daily Scrum Meeting. En annan viktig aspekt som handlar om processen och dess transparens är att Scrum Teamet inte får visa upp inkomplett funktionalitet för Product Owner under Sprint Review Meeting. Då kan Product Owner bli felinformerad om aktuell status, och på det viset undergrävs hans möjlighet att prioritera nästkommande sprint korrekt (Schwaber 2007).

Det handlar även till stor del att teamet måste lära sig självorganisering och självbestämmande, som nämnts tidigare. Det innebär också att teamet accepterar och hänger sig åt att nå de för sprinten uppsatta målen. Detta är teamets huvudfokus och teamet måste inse betydelsen som ligger i att leverera det som de har gått med på att producera. Sprintmål kanske inte kan nås alltid på grund av missförstånd, felaktigt uppskattad komplexitet, naturkrafter etc. Men teamet får inte gå på gång misslyckas med att leverera uppsatta mål, det kan skapa en dålig arbetsmiljö och självkänsla hos teamet (Schwaber & Beedle 2002).

Efter en avklarad sprint har teamet möjlighet att anpassa vilka tekniker de ska använda i nästkommande sprint, tekniker som de anser krävs för att de ska kunna nå uppsatta mål (Schwaber 2004) och inte går stick i stäv med Scrumprocessen (Schwaber & Beedle 2002).

## 3 Metod

---

*Metod-kapitlet som följer ämnar att förklara hur vi genomfört vår studie (Både övergripande och i detalj), vilka metoder för datainsamling vi använt oss av, och hur dessa data sedan har använts, med hänsyn till både praktiska och etiska aspekter.*

---

### 3.1 Forskningsstrategi

Vi har arbetat med ett kvalitativt angreppssätt, och det var för att kunna få, som Bryman (1997, 2002) beskriver det, en detaljerad bild av problemområdet, samt att komma till kärnan i det som Bryman (1997, s77) beskriver:

*”Det mest grundläggande draget i kvalitativ forskning är den uttalade viljan att se eller uttrycka händelser, handlingar, normer och värden utifrån de studerade personernas egna perspektiv.”*

Detta skiljer sig från den kvantitativa forskningen, vars fokus snarare ligger på insamlade av statistisk data för att bevisa en teori eller hypotes (Bryman 1997).

Vi var redan från start ganska klara med vad vi ville undersöka och inom vilken domän detta skulle ske. Det ledde oss till att undersöka en händelse som hade hänt i en given kontext och för det valde vi fallstudie som forskningsstrategi. Yin (2002) definierar en fallstudie som en empirisk undersökning som undersöker ett samtida fenomen i dess naturliga miljö, speciellt när gränserna mellan fenomen och kontext inte är helt uppenbara.

Tyvärr kunde vi inte få tag på organisationer som var i startgropen med att föra in Scrum med XP-relaterade tekniker, utan vi fick kontakt med organisationer som redan hade genomfört det initiala arbetet med en sådan, processen var redan igång inom utvecklargrupperna. Om man ser en Scrum-implementering som en statisk händelse som sker en gång kan man vid första anblick anta att det måste studeras i ett initialt skede. Men då fallet inte hade använt Scrum speciellt länge (4-5 månader) och Scrum är en empirisk process som formar sig efter varje iteration, ansåg vi att det passade våra ändamål bra. Till det kommer också att graden av erfarenhet av agila metoder i den undersökta systemutvecklargruppen inte var speciellt hög, vilket stärker valet av fall ytterligare.

Vi valde att genomföra en ensam fallstudie, och valde utifrån Yin (2002) att forma den efter vad han kallar ett representativt eller typiskt fall. Detta betyder följande (Yin 2002, s.41):

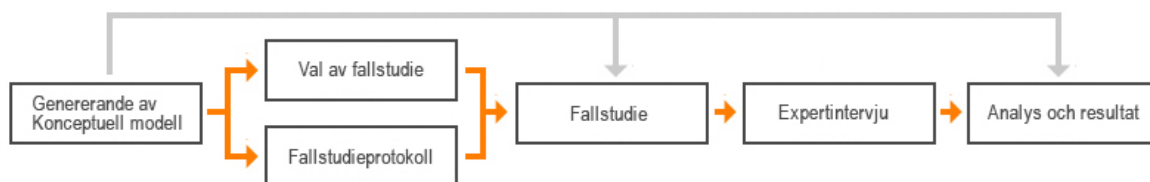
*Here, the objective is to capture the circumstances and conditions of an everyday or commonplace situation. The case study may represent a typical “project” among many different projects [...]*



Med detta i åtanke valde vi att ytterligare inrikta oss på ett fall där det inte förekom alltför extrema situationer. Vi valde ett ensamt fall på grund av att vi ville få större möjlighet att gå ner på djupet av det aktuella fallet. Samtidigt var vi oroliga för risken att tappa fokus om vi fokuserade på mer än ett fall.

Vi skapade en forskningsdesign för vår studie, något som Yin (2002) förklarar som en ritning över hur man ska bedriva sin forskning. En ritning som ska hantera fyra problem; vilken fråga ska studeras, vilken data är relevant, vilken data ska samlas in och hur ska man analysera resultaten? Först och främst är forskningsdesignen till för att undvika situationer där bevisen inte adresserar den initiala forskningsfrågan Yin (2002). Redan i detta skede valde vi publik för vår fallstudie, vilket enligt Yin (2002) är viktigt för att man ska hålla sig på rätt linje under forskningens gång.

För att få en större förståelse över området studerade vi initialt litteratur inom områdena agila systemutvecklingsmetoder, Scrum och gruppsykologi. Vi konstruerade utifrån detta en konceptuell modell (som återfinns senare i kapitel 3.2) över vad som var de övergripande punkterna utifrån vår forskningsfråga. Vi var i behov av denna modell för att hålla oss på rätt spår under vår forskning. Yin (2002) beskriver att man utöver sin forskningsfråga kan använda sig av en typ av påståenden i sin forskningsdesign för att minska risken för att hamna på villospår, vår modell fungerade som detta. Det är också en viktig punkt i att utvidga sitt teoretiska ramverk menar Yin (2002). Det hjälpte oss även att få fram vilken data som är relevant att samla in, då modellen användes som grund för intervjuguiderna. Den konceptuella modellen förändrades allt eftersom vi lärde oss mer om ämnet och vi använde den som frågeunderlag under vår fallstudie. Efter fallstudien genomfördes en expertintervju med Scrumgrundaren Jeff Sutherland, som kretsade kring det vi fann i fallstudien. Denna intervju genomfördes på grund vi saknade svar gällandes vad Scrum ansåg om viss data vi samlat in. I vår analys och resultatdel jämförde vi sedan vår konceptuella modell med resultaten av fallstudien. Vår forskningsstrategi summeras i bild 3.1, Forskningsstrategi.



Figur 3.1, Forskningsstrategi

Vår undersökning bygger på ett deduktivt synsätt, då vi bygger upp en modell bestående av kausala samband och relevanta faktorer, för att sedan utsätta denna för empirisk prövning. Dessa samband och faktorer kretsar alla kring vår analysenhet (Eller "Unit of Analysis" som Yin (2002) benämmer det), som för oss är utvecklargruppen i Scrumprocessen.

Inför datainsamlingen tog vi fram ett fallstudieprotokoll för vår fallstudie (se Bilaga 1). Fallstudieprotokollet användes för att guida oss när vi genomförde intervjuerna och som en påminnelse till oss själva vad vi skulle fokusera på. Vi utformade det efter rekommendationerna från Yin (2002) vilket innebar att det innehöll en överblick av fallstudien, beskrivning av fältprocedurer och hur vi fick åtkomst till företaget, vår konceptuella modell fungerade som fallstudiefrågor och vi tog även med vår övergripande forskningsfråga och slutligen en guide för fallstudierapporten.

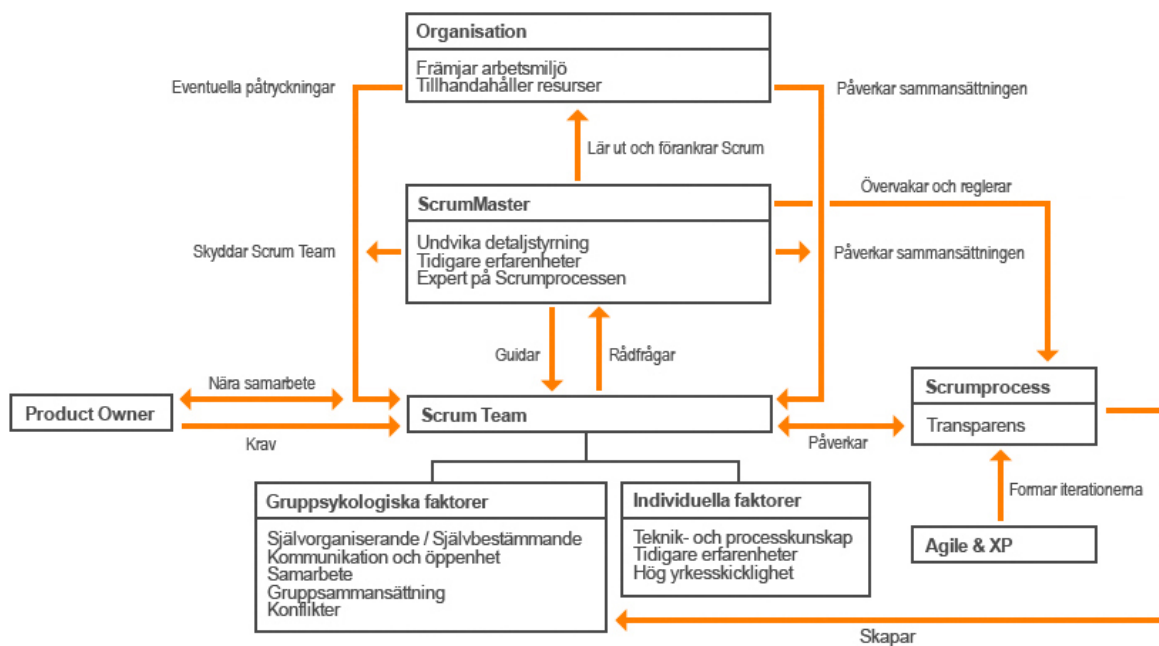
När det gäller att positionera sig själv inför sin forskning, beskriver Creswell (2007) flera (i hans ord) paradig eller världssyner, som han anser att man bör ta ställning till. En av dessa är den postpositivistiska världssynen, och är den vi anser passar bäst in på vår specifika forskning, med dess uppfattningar om hur forskning bör ske. Vad detta innebär för oss sammanfattas väl av Creswell (2007, s.20):

*”In terms of practice, postpositivist researchers will likely view inquiry as a series of logically related steps, believe in multiple perspectives from participants, rather than a single reality, and espouse rigorous methods of qualitative data collection and analysis. They will use multiple levels of data analysis for rigour, employ computer programs to assist in their analysis, encourage the use of validity approaches, and write their qualitative studies in the form of scientific reports, with a structure resembling quantitative approaches (e.g. problem, questions, data collection, results, conclusions).”*

Till att börja med så är det flera faktorer av detta påstående som passar uppbygget för en fallstudie, så som att steg för steg arbeta sig fram till ett eventuellt resultat, samt att använda noggranna och rigorösa metoder i sin analys (Yin 2002; Creswell 2007). Vidare så passar strukturen som Creswell (2007) lägger fram, det som vi lärt oss vara en praktisk och logisk följd att hålla sig till i en forskningsundersökning, en följd som är lätt att följa och förstå.

### 3.2 Konceptuell modell

Nedan presenteras den konceptuella modellen som föregående kapitel resulterade i. Modellen guidade oss genom hela uppsatsen och var således en viktig del av vår metod. Vi försökte hela tiden att revidera den allt eftersom vår kunskap om undersökningsfrågan utökades.



Figur 3.2, Konceptuell modell

### 3.3 Urval

Vi valde företaget (som beskrivs senare i kapitel 4.1, fallbeskrivning) och avdelningen på grund av att de inte tidigare hade arbetat efter Scrum eller agila tekniker. Hade vi valt företag som redan arbetade agilt hade det troligtvis inte resulterat i lika många intressanta observationer vid införandet av Scrum, något som stöds av (Schwaber 2007). Konsekvenserna av ett införande av Scrum blir således tydligare. Informanterna valdes ut i förhållande till rollerna i Scrum. Vi pratade med linjeföraren, ScrumMaster, Product Owner, samt fyra medlemmar ur Teamet. Vi hade ursprungligen som mål att intervjua alla medlemmar i Scrum Team, men efter konsultation med ScrumMaster valdes medlemmarna ut efter vilka som var mest relevanta för fallet. Faktorer som spelade in var bland annat att få informanter med olika lång yrkeserfarenhet.

Vi kom i kontakt med företaget genom att vi hörde med kollegor och bekanta om de kände till några personer som arbetar eller har arbetat med att föra in Scrum på sin arbetsplats. En bekant kände till en konsult som arbetade på det aktuella företaget, vilket var linjeföraren på avdelningen, vi tog sedan kontakt med honom via e-post.

### 3.4 Datainsamling

#### 3.4.1 Förberedelser för insamling av data

Baserad på vår konceptuella modell tog vi fram en intervjuguide (som återfinns i bilaga 1) i linje med rekommendationerna från Kvale (1997) och Bryman (2002). Det tog sig formen av fyra stycken intervjuguides, en per roll i Scrum och en för linjeföraren. I dessa intervjuguides fokuserade vi på frågor relaterade till de gruppsykologiska faktorerna men även på förändringar i det dagliga arbetet orsakade av Scrum och agila tekniker. En intervjuguide är bra att använda för att hålla fokus på den aktuella fallstudien. Intervjuerna får inte riskera att sväva ut från huvudämnet allt för mycket (Bryman 2002; Kvale 1997). I vårt fall fokuserade vi frågorna till stor del på de gruppsykologiska faktorerna som påverkar gruppen samt eventuella förändringar i sättet att arbeta.

Yin (2002) menar att det krävs noggranna förberedelser för insamling av data. Bland annat ska den som undersöker ha insikt i det ämne som undersöks, vilket vi anser vi fick genom framtagandet av vår konceptuella modell. Vidare ska han även vara en bra intervjuare med förmåga att fråga rätt frågor och vara en bra lyssnare. Detta menar Kvale (1997) är ett hantverk som man kan nå via erfarenhet och observationsförmåga.

#### 3.4.2 Insamling av data

Som metod för att samla in vår empiriska data valde vi i huvudsak att använda oss av semistrukturerade intervjuer, i enlighet med ett kvalitativt arbetssätt (Kvale 1996; Bryman 2002). Att använda sig av en sådan struktur medför en rad fördelar som Bryman (2002) redogör för. Tack vare att frågorna är öppnare tillåter de informanterna att uttrycka sig friare, och ger utrymme till konstruktiva följdfrågor samt att man får möjlighet att utforska nya ämnen som uppkommer under själva intervjusituationen. Frågorna i en intervju av detta

slag lämpar sig dessutom att vara korta och koncisa, för att undvika att styra informanten allt för mycket, eller att lägga ord i hans eller hennes mun (Kvale 1996).

Intervjuerna genomfördes öga mot öga, något som Bryman (2002) anser har både för- och nackdelar. Det blir en mer personlig intervju, och det uppstår ett slags samarbete mellan intervjuare och informant. Till exempel så kan informanten förstå när vissa ämnesområden behöver utvecklas vidare baserat på intervjuarnas ansiktsuttryck, ställning och gester (Holme & Solvang 1997; Bryman 2002; Patel & Davidson 1994). Kvale (1997) bygger på denna uppfattning med tillägget att intervjuaren samtidigt ska agera intresserad och förstående för det som informanten delar med sig av, och att visa respekt för denne. Intervjuerna spelades in på bandspelare för att sedan kunna transkriberas och sedermera analyseras. Före själva intervjustarten informerade vi informanten om syftet med vår undersökning och att denne hade all rätt att vara anonym, vilket ingen person valde. Dock spelades inte expertintervjun in på band samlades in genom anteckningar från intervjun.

Vidare undersökte vi och lokaliserade interna dokument inom organisationerna, dokument som berörde Scrum och organisationernas systemutvecklingsprocesser. Detta för att få möjligheten att använda flera källor som bevis. Insamlingen av dessa skedde via linjeföraren, och överfördes elektroniskt till oss.

Vi utförde även direktobservation av två stycken möten direkt kopplade till Scrum-processen, nämligen Daily Scrum Meeting. Yin (2002) beskriver dessa som ytterligare källor av data i en fallstudie, och belyser deras användbarhet i att få se olika sidor av fenomenet man studerar. I enlighet med Yin (2002) genomförde vi dessa observationer tillsammans istället för individuellt, för att öka reliabiliteten på de resultat vi fick fram. Vi var inte delaktiga i mötena utan endast passiva observatörer. Medan vi gick omkring på avdelningen försökte vi även vara observanta på interaktionen mellan de anställda och vad de hade för sig. Dessa observationer över vad som utspelade sig i arbetslokalerna tecknades ned efter att vi hade avslutat aktuellt besök, och har återgetts i bilaga 7.

Viktigt att belysa är att ungefär samtidigt som vi genomförde intervjuerna genomfördes en omorganisation och Scrum Teamet splittrades. Det delades in i två team och nya människor kom in i teamen. De hade dock inte börjat arbeta i de nya teamen utan var fortfarande i den sista sprinten med den första formeringen. Eftersom vi ville hålla vår analysenhet intakt arbetade vi aktivt med att fokusera oss på det som hände innan teamet splittrades, detta var något som vi försökte fokusera på under alla intervjuer. Detta medförde således även att vi inte kunde medverka på vissa möten, exempelvis vissa planeringsmöten, då dessa möten genomfördes med den nya indelningen. Att fokusera på rätt analysenhet är något som Yin (2002) menar är svårt men väldigt viktigt under en fallstudie och hade således detta i åtanke.

### *3.4.3 Negativa instanser*

I vår analys av det empiriska material vi samlat in har vi gjort vårt bästa för att hitta och uppmärksamma de tillfällen då vi hittat data som motsäger varandra, eller ”negativa instanser” som bland annat Seale (1999) kallar dem. De negativa instanserna tas även upp i vår diskussion.

Seale (1999) menar att det finns flera anledningar till detta, till exempel för att ge röst åt de åsikter som annars inte hade blivit hörda, att bevisa att ens resultat inte är alltför vinklade

och ensidiga, samt att låta dem influera ens slutliga resultat. Vi hoppas därmed att på detta sätt kunna dra nytta av detta förfarande.

### 3.5 Analys av data

De inspelade intervjuerna transkriberades av oss i enlighet med Bryman (2002). De praktiska fördelarna med att göra detta är bland annat att man som forskare har möjligheten att se över sina resultat på ett enklare sätt, och slipper att lyssna igenom intervjusamtalen gång på gång för att försäkra sig om att man tolkat ett visst område korrekt. Transkriberingarna utgör dessutom en källa för läsare att få en inblick i studien på ett bättre sätt, och det går att se huruvida forskarnas analys speglar verkligheten (Bryman 2002).

Transkribering är dock inte helt utan problematik. Förutom att det är tidskrävande (ett problem som är relativt lättöverkomligt) så skapas frågan om huruvida det ska göras ordagrant eller om en viss omskrivning ska tillåtas. Talspråk kan vara mindre smickrande för informanterna i efterhand, och frågan är än mer relevant i de fall då deras sanna identitet inte dolts, då informanterna då kan tycka att deras talspråk inte förmedlar deras personliga åsikter och erfarenheter på ett korrekt sätt (Bryman 2002; Kvale 1997).

Vi valde att skriva om intervjuerna till mer korrekt svenska, och lade ner extra energi på att se till att inte skriva om något stycke avgörande för kontext eller mening. Kvale (1997) belyser just detta och stödjer omskrivande så länge som det inte ändrar innebörden i budskapet. En ordagrann återgivning är något som bättre lämpar sig för en språklig analys av intervjuerna (Kvale 1997). För att försäkra oss om att vi uppfyllde alla kraven för en pålitlig översättning från ljud till ord, bemödade vi oss med att läsa igenom varandras transkriberingar i samband med att vi lyssnade på intervjuerna, för att försäkra oss om att inga allvarliga misstag smugit sig in, något som kan ha allvarliga konsekvenser för analysen (Bryman 2002).

Före själva analyserandet av vår empiriska data valde vi att koda informanternas svar enligt kategorier vi arbetade fram. Vi gick tillväga som så att vi tog vår konceptuella modell och formade en mall av kodningskategorier (bilaga 3), varpå vi satte igång att individuellt koda intervjuerna, och sedan att gå över och korrigera varandras kodning. Detta för att inte påverkas av varandras idéer. Efter hand som kodningen framskred, upptäckte vi fler kategorier, och införde dem allteftersom vi stötte på dem. Ett exempel är att de intervjuande ansåg att det kollektiva ansvaret hade ökat sedan de införde Scrum. Vi upptäckte även efteråt att ett par av våra kategorier verkade irrelevanta, varpå vi strök dem.

Detta arbete resulterade i dels en mall över de olika slutliga kategorierna, och dels en stor sammanställning över alla kommentarer och svar som informanterna gav inom de olika områdena. I bilaga 4 återfinns några exempel från vår sammanställning.

Resultatet av denna kodning och av vårt förfinande av kategorierna ledde till skapandet av en ny konceptuell modell, baserad på de nya kategorierna och därmed annorlunda än vår ursprungliga modell. Att ha två modeller tillät oss då att genomföra vad Yin (2002) kallar ”pattern matching”, genom att jämföra ett förutspått mönster (en modell) med ett empiriskt framarbetat sådant.

## 3.6 Etik och forskningskvalitet

### 3.6.1 Övergripande etiska ställningstaganden

De etiska implikationerna är många när man genomför en kvalitativ undersökning med intervjuer som den främsta källan till ens empirisk data, och det är viktigt att de beaktas och hanteras på ett adekvat sätt. Det finns många potentiellt skadliga fallgrorpar (Sieber 2001; Singer & Winson 2002; Israel & Hay 2006).

En god utgångspunkt specificeras av Israel och Hay (2006) i att målet med forskning alltid bör vara att skydda andra, minimera skada, samt att öka de positiva effekterna av ens forskning. För vår del innebar detta en rad förebyggande åtgärder, bland annat informerat samtycke, konfidentialitet, fördomar och konsekvenser.

Vi införskaffade informerat samtycke från varje informant innan det att han eller hon intervjuades, och detta skedde via skriftliga intyg. Denna praxis stöds både av Creswell (2006) och Israel och Hay (2006). Å andra sidan kan ett sådant intyg göra att intervjuprocessen ses som mer byråkratisk och ogästvänlig, vilket minskar de kvalitativa värdena hos den mer informella intervjuformen (Israel & Hay 2006). Ansvar läggs då på forskarna som måste hantera detta och därmed åstadkomma en god intervjusituation.

För att komma till kärnan av vad informerat samtycke är, kan det vara på sin plats med ett citat från Singer & Winson (2002, s.3):

*“Ethicists do not fully agree on the necessary components of informed consent, but it is clear that it must contain at least some of the following elements: disclosure, comprehension and competence, voluntariness, the actual consent or decision, and the right to withdraw from the experiment.”*

Detta påstående innehåller en rad element som är viktiga för oss. ”Disclosure” och ”comprehension” innebär att informanterna är informerade om forskningsstudiens syfte och omfattning i förväg, samt att de förstår vad detta innebär för deras egen skull. Vi löste detta genom att lägga ner lite extra tid och energi i samband med intervjuerna för att försäkra oss om att våra informanter hade denna förståelse. Kvale (1996) kallar detta ”orientering”, och föreslår att det sker både före och efter intervjun, för att fånga in eventuella oklarheter och stärka informantens förtroende för att hans eller hennes deltagande hanteras på ett propert och etiskt sätt.

En viktig aspekt av informerat samtycke som belyses av Singer och Winson (2002) är att informanter ibland utsätts för organisatoriska påtryckningar för att delta. Ett exempel på detta är när en projektledare ställer sin arbetsgrupp till förfogande, och därmed förväntar sig att de ska ställa upp (Singer & Winson 2002). Detta var högst aktuellt för oss, och hanterades genom den initiala kontakten med informanterna, då vi gjorde vårt bästa för att försäkra oss om att detta inte var fallet. Vidare så klargjorde vi för informanterna att deras medverkan var helt frivillig, samt att de hade möjlighet att dra sig ur när som helst. För att försäkra oss om att detta skedde etiskt så krävdes ingen förklaring vid ett sådant avhopp (Singer & Winson 2002).

Singer och Winson (2002) beskriver dessutom ytterligare ett problem i periferin av informerat samtycke, nämligen att de som ger det måste vara kompetenta nog att förstå

konceptet bakom samtycket. Exempel på sådana situationer är när minderåriga barn eller mentalt handikappade ska intervjuas. För vår del handlade det istället om att använda terminologi och språk som informanterna förstod och lätt kunde ta till sig, och inte presentera dem med ett byråkratiskt dokument liknande en lagtext.

Nästa viktiga steg är konfidentialitet, som konsekvent framhävs i litteratur som en viktig komponent inom etisk kvalitet, bland annat av Bryman (2001), Israel och Hay (2006), Kvale (1996) och Creswell (2006).

Vad gäller just konfidentialitet så finns det två extremer att ta ställning till. Å ena sidan finns behovet att skydda informanterna från att deras konfidentiella uppgifter blir offentliga. Detta kan ske genom att individer insatta i informantens arbetskontext kan se igenom eventuella försök att dölja dess identitet (Israel & Hay 2006). Å andra sidan tar Kvale (1996) upp problemet med att dölja för mycket, då viktiga samband och kontexter kan gå förlorade.

Vår lösning på detta problem var att inte utgå ifrån hundra procentig konfidentialitet, utan istället fråga informanterna individuellt till vilken grad de skulle kunna identifieras. Israel och Hay (2006) belyser detta och menar att det inte heller är alla som vill förbli konfidentiella. Alla presenterades dock för alternativet att förbli helt anonyma.

Även konsekvenserna för ens forskning är högst viktiga att beakta, både när det gäller informanterna och företagen inom vilka de arbetar.

### *3.6.2 Reliabilitet*

Reliabilitet innefattar möjligheten att återskapa fallstudien i det mening att om en annan forskare genomför exakt samma fallstudie, ska han nå samma resultat och slutsatser (Yin 2002). Det innebär i praktiken att målet med reliabilitet är att en forskare ska kunna genomföra vår fallstudie i samma kontext och finna det vi fann.

Vi skapade och använde ett fallstudieprotokoll för vår fallstudie vilket enligt Yin (2002) är en av de viktigaste punkterna när det kommer till att öka reliabiliteten i en undersökning. Yin (2002) ger tipset om att man hela tiden ska ha detta i åtanke, vilket vi försökte ha under hela vår forskningsprocess.

### *3.6.3 Validitet*

Validitet behandlar korrektheten i en undersökning. I praktiken innebär det att man återspeglar verkligheten på ett korrekt sätt, och att man fokuserar på att undersöka rätt saker utifrån sin forskningsfråga. Då vi har intervjuat flera personer i vårt fall har vi försökt få en så rik bild av det undersökta problemet som möjligt.

Ytterligare ett steg i att göra arbetet pålitligt var att vi genomförde respondentvalidering genom att vi lät våra informanter ta del av deras respektive transkriberingar, i hopp om att fånga upp potentiella feltolkningar, det kan också vara så att intervjupersonen upptäcker felaktigheter i sitt svar som han vill korrigera. Tack vare att vi gjorde detta lyckades vi fånga upp ett par fel, där informanterna insåg att deras svar ord för ord inte helt överrensstämde med vad de faktiskt tyckte, vilket fick oss att korrigera intervjutranskripten.

Validitet går i sin tur att bryta upp i en rad deltyper, av vilka vi kommer att ta upp intern validitet och konstruktvaliditet. Intern validitet beskrivs som följande av Bryman (2002, s44):

*”Intern validitet handlar om huruvida en slutsats som rymmer ett kausalt förhållande mellan två eller flera variabler är hållbart eller ej. Om vi påstår att x orsakar y, kan vi då vara säkra på att det verkligen är x som svarar för variationen i y och inte någon annan faktor [...]”*

Yin (2002) bygger vidare på detta genom att skriva att intern validitet inte är lika viktig för en utforskande studie, men att den interna validiteten ökar genom användandet av mönstermatchning, som tidigare nämnts.

Betydligt mer relevant och viktigare är då konstruktvaliditet, som beskrivs som följande av Yin (2002, s35):

*“Construct validity: establishing correct operational measures for the concepts being studied.”*

Vad detta betyder är att det som ska mätas inte alltid stämmer överens med verkligheten, och att undersökare bör vara försiktiga med att vara för subjektiva i samband med dessa mätningar.

För att uppfylla kraven för konstruktvaliditet nämner Yin (2002) tre principiella åtgärder. Först av allt, så är det fördelaktigt att använda sig av multipla källor för sitt empiriska material, vilket vi gjort genom att inte endast fokusera på intervjuer, utan även utföra direktobservationer och undersöka interna dokument.

Steg efter det är att etablera vad Yin (2002) kallar en beviskedja, genom att göra det tydligt från vilka delar av empirin som analys och slutledningar härstammar, och att göra dessa tillgängliga för läsaren. Genom att i vår analysdel flitigt notera ifrån vilka intervjuer eller observationer vårt material kommer, och genom att bifoga vårt fallstudieprotokoll försöker vi uppfylla detta krav.

Det tredje och sista steget för att nå konstruktvaliditet är att låta informanterna själva se över den färdiga rapporten, för att låta dem komma med rättelser och ändringar där feltolkningar skett, m.m. Yin (2002) menar att detta steg inte endast är professionell hövlighet, utan även är ett viktigt steg i att öka kvaliteten och konstruktvaliditeten. En av anledningarna till detta är att våra egna eventuella subjektiva åsikter kan korrigeras och tas bort. Efter att ha slutfört vår analys och våra slutsatser delade vi med oss av vårt färdiga resultat till informanterna. Ytterligare en punkt i detta steg som genomfördes var att vi lät informanterna ta del av intervjutranskripten innan vi började med analysen. Detta för att ge dem ytterligare möjlighet att korrigera eventuella fel, samt om möjligt ge extra förklaringar till något vi undrat över under transkriberingen. Resultatet av detta var ändringar av mindre detaljer där inspelningen varit otydlig eller smärre missförstånd skett.

### 3.6.4 Bias och dess hantering

Att så gott som all forskning påverkas av bias (t.ex. i form av förutfattade meningar) är en åsikt som delas av flertalet författare, bland dem Ehrlinger et al (2007), Hammersley och Gomm (1997) samt Norris (1997). Ehrlinger et al (2007) skriver att bias är betydligt svårare



att upptäcka (och därmed hantera) i sig själv än i andra. Hammersley och Gomm (1997) vidareutvecklar detta med att göra en distinktion mellan medveten och omedveten bias, vilket ytterligare försvårar identifierandet.

Ett av tillvägagångssätten vi använt oss av för att undvika bias är det som tagits upp tidigare i delkapitlet om validitet, nämligen att vi använt oss av respondentvalidering och låtit våra informanter ta del av intervjutranskript och av uppsatsen som helhet. Ifall vi hade haft bias är det vår tro att denna validering minskat effekterna av biasen.

Ytterligare stöd fås via våra metoder för insamling, t.ex. att vi genom triangulering och sökandet i multipla datakällor förhoppningsvis blir tvingade att komma till slutsatser som inte är alltför färgade av bias.

### *3.6.5 Generalisering*

Yin (2002) och Flyvbjerg (2006) skriver att det existerar kritik mot fallstudier att det inte går att generalisera ifrån deras resultat, och att de därför skulle ha ett lägre vetenskapligt värde.

Att så inte skulle vara fallet argumenterar de bägge två för. Yin (2002) anser att även om en fallstudie är långt ifrån ett statistiskt urval, så går det fortfarande att göra generaliseringar på teorinivån. Precis som med resultaten av ett vetenskapligt experiment, går det att ifrån en fallstudie dra slutsatser och finna teorier som är applicerbara utanför fallets kontext (Yin 2002). Flyvbjerg (2006) expanderar på just detta och tar upp falsifiering som ett primärt exempel där fallstudier kan resultera i generaliserbara slutsatser. I hans exempel är syftet att motbevisa teorin ”alla svanar är vita”, och med ett enda fall går detta att åstadkomma. Han anser vidare att detta koncept går att översätta till fallstudier, vilket skulle innebära att deras vetenskapliga värde ökar (Flyvbjerg 2006).

### *3.6.6 Källkritik*

Det går inte att komma ifrån det faktum att Scrum är en kommersiellt paketerad produkt. Certifieringar inom Scrum finns att få via många utbildningsföretag, även grundarna själva håller i utbildningar inom området. Vi har försökt ha detta i åtanke under vår uppsats och letat efter kritiska röster gentemot Scrum, men inte hittat några med vetenskaplig tyngd. Vi har sökt i vetenskapliga databaser så som JStor, ScienceDirect, ELIN@Lund (Lunds Universitets Biblioteks elektroniska resurser), Emerald Insight och ACM Digital Library.

Att vi inte hittat kritiska röster till Scrum bland akademiska publikationer kan ha flera förklaringar, till exempel att det är relativt nytt, och att förståelsen för effekterna (speciellt de långsiktiga) av Scrum inte har nått den grad att det resulterat i den sortens kritik som vi sökt efter.

Det existerar kritik kopplad till agilitet, men mycket av denna har redan hanterats i de agila förhållningssätten i form av argument gällandes att de scenarion som kritiker målar upp är missriktade, och att det i de problematiska situationerna inte är agilitet som är problemet, utan något annat. XP har fått kritik bland annat i boken *Extreme Programming Refactored: The Case Against XP* (Stephens & Rosenberg, 2003) men vi ansåg inte att kritiken i denna bok gick att applicera på vår undersökningsfråga, syftet med uppsatsen var inte att utvärdera agilitet och dess kritik.

## 4 Resultat och analys

---

*I detta kapitel presenteras vår fallstudie och resultatet av denna. I slutet presenteras en modell över resultatet och en förklaring av denna modell.*

---

### 4.1 Fallbeskrivning

Företaget där fallstudien genomfördes på var SonyEricsson (SEMC) i Lund och avdelningen Test Automation Development (BGLVSB). Avdelningen hör till Global Product Verification sektionen på SEMC. Avdelningen har funnits sedan maj 2007 och består av 12 anställda, de anställda består av fyra fastanställda, samt åtta inhyrda konsulter. Avdelningen utvecklar verktyg för automatiserade tester av mobiltelefoner som används internt av SEMC på dess olika Global Test Centers. Den största applikationen som utvecklas på avdelningen heter Batch Remote Automated Testing (BRAT), den simulerar hur en användare använder en mobiltelefon. Utöver det driver avdelningen en del småprojekt inom området. Produkten BRAT började utvecklas hösten 2005 av nuvarande linjeförman Anders Nyberg och en kollega till honom när dessa arbetade som testare på SEMC. Det började som ett hobbyprojekt, men under mitten av 2006 tog projektet fart och nuvarande linjeförman blev projektledare. Produkten växte i storlek och blev ett officiellt projekt och avdelningen bildades. Det är användarna av produkterna som i huvudsak är de personer som lägger krav på ny funktionalitet och förbättringar av produkterna.

#### 4.1.1 Avdelningen och respondenter

Avdelningen är styrd av linjeförman Anders Nyberg och under honom finns det 11 anställda som arbetar med utveckling av testautomationsverktyg. Under vår fallstudie genomförde vi sju intervjuer, vi intervjuade Product Owner, ScrumMaster, linjeförman samt fyra medlemmar av ScrumTeamet. Nedan följer en kort presentation av våra respondenter.

Anders Nyberg är linjeförman över avdelningen Test Automation Development och är en av personerna som var med och tog fram huvudprodukten BRAT från start, och har således en bakgrund som utvecklare. Han har arbetat på SEMC sedan september 2004, först som konsult, men 2007 när avdelningen bildades gick han över till att bli anställd på SEMC.

Niklas Holmberg är den som agerar Product Owner för produkten BRAT, han var tidigare projektledare för produkten. Han har arbetat med BRAT sedan våren 2005, först som underkonsult, men blev i april 2007 anställd på SEMC. Niklas har en bakgrund som interaktionsdesigner och är utbildad på Malmö Högskola och har före detta projekt ingen erfarenhet av agila tekniker eller Scrum.

Lars Ahlkvist är avdelningens utbildade ScrumMaster och har arbetat på SEMC sedan 2003, Lars är i grund och botten utvecklare och har arbetat med systemutveckling sedan 1995. Hans roll inom avdelningen innefattar även en utvecklarroll i ScrumTeamet och han började använda agila tekniker så som TDD i mitten av 2007.

Jonas Nyberg är ScrumTeam-medlem och har arbetat som konsult på SEMC och BRAT-projektet sedan juni 2007. Han har arbetat med systemutveckling sedan ungefär 1997 och har innan detta projekt inga erfarenheter av agila tekniker, men stor erfarenhet av vattenfallsliknande förhållningssätt.

Martin Sternebring är ScrumTeam-medlem och började arbeta på avdelningen och BRAT-projektet i oktober 2007. Han är inhyrd underkonsult och har arbetat med systemutveckling sedan 2004, främst teknisk projektledning och arkitektur. Martin har sedan tidigare viss erfarenhet av agilitet, då han arbetat i ett projekt med mycket täta kundkontakter och en form av iterativ systemutvecklingsmetod där kraven förändrades ofta.

David Jönsson är medlem i ScrumTeamet och har arbetat som inhyrd konsult på avdelningen sedan den bildades. David har arbetat på SEMC sedan mars 2006, han började som skriptutvecklare för BRAT men har sedan september 2007 en systemutvecklroll. David har även det övergripande ansvaret för release och versionshanteringsfrågor på avdelningen. Före BRAT-projektet hade han ingen erfarenhet av agila tekniker eller Scrum utan han har arbetat efter olika vattenfallsbaserade systemutvecklingsmetoder. Hans roll innefattar även att vara ScrumMaster för ett testprojekt mot en avdelning i Kina.

Karl Eklund har arbetat som medlem i ScrumTeamet sedan hösten 2007, han har dock varit anställd som underkonsult på SEMC sedan april 2005, innan det läste han informatik på Lunds Universitet. Innan han blev medlem av BRAT-projektet arbetade han i en testgrupp som testade mobiltelefoner åt SEMC. Karl har före detta projekt ingen arbetslivserfarenhet av systemutvecklingsmetoder eller agila tekniker. Hans arbete har tidigare handlat om Use Case och Test Case framtagning men han ska nu under våren 2008 ha en roll som systemutvecklare.

#### *4.1.2 Scrum på avdelningen Test Automation Development*

Scrum började införas på avdelningen någon gång under augusti, september 2007 på initiativ av Anders Nyberg. En sprint sträcker sig över en tvåveckorsperiod, detta för att de anser att fyraveckorsperioder skapar för långa väntetider på att förändringar i applikationen ska skeppas ut till organisationen.

Tidigare hade de arbetat ganska agilt i form av att de satt nära de som ställde kraven och hade ett bra samarbete med dessa. När de sedan flyttade ut till en egen avdelning började de märka att de förlorade kontakten med dessa människor och det blev svårt att veta vad de egentligen ville ha. Efter att Anders Nyberg hade varit på ett föredrag om Scrum började de studera litteratur inom området, ett Scrumteam skapades för att ta hand om huvudprodukten BRAT. Detta innebar att det fanns några anställda på avdelningen som inte arbetade efter Scrum utan bedrev diverse småprojekt efter tycke och smak. Ingen hade i detta skede fått någon formell utbildning angående Scrum och dess arbetsprocess. Vid årsskiftet 2007-2008 gick Lars Ahlkvist en ScrumMaster-utbildning på Softhouse och de insåg då att de hade varit på rätt väg men missat ett par punkter i Scrumprocessen, främst kopplat till estimering.

Utbildningen inom Scrum under de första sprintarna har skett via att Anders Nyberg samt Lars Ahlkvist har hållit kortare föredrag om Scrum och sättet att arbeta. Teamet delar idag en öppen kontorsyta med övriga anställda på avdelningen samt Product Owner.

Innan Lars hade fått utbildningen agerade Anders Nyberg i viss utsträckning som ScrumMaster eftersom han som tidigare nämnts utbildade och informerade ScrumTeamet angående Scrum.

Efter vår fallstudie har de beslutat att dela upp avdelningen i två olika Scrumteam, för att de inte vill ha några som står utanför sättet de arbetar efter, samt att de tror det kommer bli lättare att hålla teamen intakta på detta sätt.

## 4.2 Utfall

Vi har här valt att presentera resultatet av våra intervjuer, insamlade dokument och direktobservationer efter de entiteter vår konceptuella modell består av. Det innebär att varje avsnitt i detta kapitel motsvarar flera koder ifrån vår konceptuella modell. Vi valde att kategorisera det efter entiteter, då vi ville få möjlighet att få med ytterligare aspekter av det vi har undersökt under respektive kategori. Aspekter som saknades i vår konceptuella modell.

### 4.2.1 Organisation

Teamet sitter tillsammans med Product Owner och ScrumMaster i ett öppet kontorslandskap. Lars Ahlkvist menar att ett stöd från organisationen krävs för att man ska kunna få Scrum att fungera fullt ut och att han känner att det stödet verkligen finns, gällande stödet förklarar han det med följande.

*[...] alla har varit väldigt öppna för det, framförallt linjeförföraren, men även organisationen utanför, vilket jag tror är ett måste. Om man kan få med linjen på det hela kan man få det att funka (Lars Ahlkvist)*

Även Jonas Nyberg, David Jönsson och Martin Sternebring påpekar att stödet från linjeförföraren är mycket starkt och tydligt. Ingen av de intervjuade anser att det är problem kopplade till att sitta i en så öppen arbetsmiljö, ”vill man lyssna så lyssnar man”, som en person uttrycker det angående att man hör mer än man kanske vill höra. Just den öppna miljön skapar kreativitet tycker Martin Sternebring och enligt honom främjas utvecklandet av arbetsmiljön av organisationen, även om det finns en teknisk vinkel på förändringarna:

*[...] det gör ju hela tiden att man tänker att man kan förbättra något, hade det inte funnits hade man förmodligen haft tråkigare på jobbet kan jag säga. Absolut. Anders har ju som har ju en dragning någon gång i månaden, varannan vecka vad det kan bli, om en ny teknik som han har hittat. Så får vi bestämma om vi tycker det är bra eller dåligt, så får vi testa det. Har vi hittat en teknik som vi tycker är nice så har vi en dragning, så det är ju öppet, ett öppet klimat får man säga. (Martin Sternebring)*

När vi behandlar frågan huruvida organisationen utövar påtryckningar på ScrumMaster och teamet så upplevde våra intervjupersoner att inga påtryckningar förekom. De tog upp att ingen utomstående utövar några påtryckningar i form av hur de ska arbeta, då dessa utomstående saknar insyn och det istället är upp till avdelningen själv att bestämma hur de ska bedriva sin verksamhet.

*Övriga har ingen insyn i vårt arbete. Skulle de cheferna ett steg upp lägga sig i, de har ju hundratals människor under sig, de skulle aldrig kunna göra micro management på det sättet. Så att de lyckas hålla fingrarna ur syltburken är ju snarare status quo, om han hade börjat med micro management, då hade vi haft ett riktigt problem (Karl Eklund)*

Den form av påtryckningar som finns inom avdelningen ville inte teammedlemmarna kalla för påtryckningar utan snarare tips. Det togs upp att det ofta var linjeföraren Anders Nyberg som kom med förslag om tekniker och sådant som de kunde använda, tekniker såsom Continuous Integration, men att de inte såg det som påtryckningar. Det poängterades dock av en teammedlem att det kanske fanns en viss press i och med att de uttalat ska använda Scrum.

*Pressen finns ju där, i och med att vi har uttalat att vi ska köra Scrum. Men jag tänker inte i daglig verksamhet om jag kör Scrum eller inte. (Martin Sternebring)*

När det gäller påtryckningar om funktionalitet på produkten finns det bara en kanal in och det är via Product Owner. Något som även påpekades av flera teammedlemmar.

*[...] men vi hänvisar hela tiden till Product Owner, vi tar ingenting som är utanför, utan vi har bara en kanal ut, och det är Product Owner. Så om någon kommer till mig så hänvisar jag till Product Owner, kommer de till teamet hänvisar de till antingen mig eller Product Owner, och jag hänvisar vidare. (Lars Ahlkvist)*

I intervjuerna togs frågor upp som behandlade vem som bestämmer och vilka som påverkar sammansättningen av teamet. Teamet har själva aldrig påverkat sammansättningen, men en teammedlem menar att de nog skulle kunna göra det om de verkligen hade velat. Teamet sattes ihop av linjeföraren Anders Nyberg och var en blandning av två team. Det första teamet bestod av personer som flyttades till avdelningen efter en omorganisation, den andra delen (60 %) bestod av personer som Anders Nyberg rekryterat, både anställda och konsulter. När teamet splittrades till två var det ScrumMaster Lars Ahlkvist och Anders Nyberg som tillsammans satte ihop det.

*Linjeföraren har satt ihop dem, jag har haft en liten del i det, och försökt att få två liknande team, så att det inte blir ett a- och ett b-lag. Det ska inte spela någon roll vilket team som arbetar med en viss produkt. De ska även ha samma kunskaper så att man sprider kunskapen inom gruppen. (Lars Ahlkvist)*

#### 4.2.2 ScrumMaster

För att Scrum ska fungera ska ScrumMaster vara expert på Scrum, det är han som förankrar och lär ut Scrum till de som är med i projekten. Eftersom Lars Ahlkvist inte initialt hade genomgått en formell utbildning angående rollen som ScrumMaster så gjordes vissa misstag i de första sprintarna. Det som tydligt märks nu är att han upplever det som att utbildningen verkligen var givande och han har stor nytta av den i sitt dagliga arbete, och på det viset kan han agera som den expert han har blivit och ska agera som. Alla teammedlemmar berättar att det är till Lars de går om de har frågor om Scrum och att han är väldigt mån om att de ska följa Scrumprocessen, han kommer med guidning och förslag på saker och ting. Även Product Owner känner stort förtroende för Lars i ScrumMaster-rollen och han kunskaper om Scrumprocessen.

*[...] och jag har stor tillförlit till ScrumMaster, han är ju utbildad och han är väldigt driven för det han gör (Niklas Holmberg)*

En sak de hade missat som Scrumprocessen kräver är att uppskatta komplexitetspoäng i Product Backlog, de hade helt enkelt feltolkat den, förklarar både Lars Ahlkvist och Anders Nyberg. Detta var något som de förändrade efter att Lars hade gått en ScrumMaster-utbildning. Vidare ändrade de också hur de planerar frånvaro (semester, utbildningar etc.) i en sprint, nu tar de med det i beräkningen från början. Innan plockade vi helt enkelt bort saker ur Sprint Backlog om en teammedlem skulle vara borta förklarar Anders Nyberg. På frågan om teamet är med och bestämmer mer nu efter att Lars har genomgått utbildningen förklarade Anders Nyberg det på följande sätt:

*Ja de var ju med och estimerade sprintarna men inte komplexiteten i Product Backlog, så visst de är ju med mer å bestämmer. (Anders Nyberg)*

Ytterligare en punkt som de i början inte visste hur de skulle hantera var den angående hur de hanterar Sprint Backlog om de inte hinner klart med alla uppgifter under en sprint. I början plockade de bara bort de saker de inte hann klart med, dagen innan sprintmålet. Vilket skapade icke rättvisande Burndown Charts, det såg ut som de hade hunnit med alla uppgifter fast de inte hade det menade linjechef Anders Nyberg.

Ytterligare information som kom fram under intervjuerna var att ScrumMaster även agerade som utvecklare i teamet, det verkade finnas lite problem kopplat till detta. På frågan om han har svårt att hålla isär sina intressen som utvecklare och ScrumMaster svarar han:

*Det kan ju bli konflikter, när man är ScrumMaster och Team-medlem samtidigt, och jag försöker så mycket som möjligt gå tillbaka och hålla mig till ScrumMaster så att teamet får bestämma utan mig. Jag tar rollen som ScrumMaster före rollen som Team-medlem.(Lars Ahlkvist)*

Han är också den mest erfarna systemutvecklaren på avdelningen, vilket gör hans roll som systemutvecklare extra betydelsefull enligt de andra teammedlemmarna. Lars upplever det ibland som folk förlitar sig på att han har en bra lösning på ett problem.

Lars upplever att hans huvuduppgift är att se till att de kan utföra sitt jobb, och han känner sig delaktig i att allt kommer i mål på grund av att han även är utvecklare i teamet. Han upplever det ibland frustrerande att ha två roller. Exempelvis när han tycker sig veta en bra lösning, och då försöka guida dem till att lyssna på hans åsikter istället för att säga till dem hur de ska göra, förklarar han. David Jönsson och Karl Eklund menar att det ibland har varit lite luddigt om när och vad man ska fråga Lars som ScrumMaster, men att det har blivit tydligare ju mer de har lärt sig om Scrum.

Alla intervjuade teammedlemmar var av uppfattningen att ScrumMaster Lars Ahlkvist inte detaljstyrde dem i deras arbete, utan mer guidade dem. Även om några upplevde det som lite svårt att hålla isär hans roller, eftersom han båda var ScrumMaster och medlem i teamet. Jonas Nyberg uttryckte sig på följande sätt:

*Det är mer guidning tycker jag, och sen har han ju koll på utvecklingen i sprinten och fångar upp om vi halkar efter eller ligger före. Men han trycker ofta på om vissa uppgifter i sprinten är extra viktiga, och att någon måste ta tag i det under dagen*

*eller senast imorgon så att vi säkert hinner med det innan sprintens slut, så det är mer på den nivån. [...] (Jonas Nyberg)*

Detta stöds även genom direktobservationerna av Daily Scrums, då vi fick intrycket av att han verkligen guidade dem i arbetet.

Angående Lars Ahlkvists dubbla roller och att han är den som i teamet besitter störst kunskap om produkten och om systemutveckling, beskriver han att det är viktigt att han inte säger åt någon i teamet vad de ska göra om de kommer med en fråga. Han menar att han måste guida dem och lägga fram hur man kan lösa saker och ting. Skulle han säga åt dem exakt vad de ska göra fungerar det inte alls bra menar han. Tidigare hade ju även Anders Nyberg en form av ScrumMaster roll och den verkar i viss mån finnas kvar, eller snarare att han föreslår en teknik teamet kanske kan använda sig av, och så utvärderar de denna teknik. På frågan vem det är som pushat på användandet av Continuous Integration och tekniker de använder svarar en av de intervjuade på följande sätt.

*Det är Anders som nog stått för det mesta. Det har ju kommit fram idéer och så från oss, men jag har uppfattat det som att det är han som driver det mesta framåt så. (David Jönsson)*

I vår intervju med Anders Nyberg kom det på tal att han har fört in förslag på hur teamet ska arbeta i form av vilka tekniker de ska använda sig av och om detta inte innebär att han således detaljstyr teamet. Anders förklarar det genom att han är utvecklare i grunden och har ett intresse för att avdelningen ska använda och testa de tekniker som kan hjälpa dem att få en bättre produkt. Men att han inte säger åt dem att de måste använda en viss teknik utan mer tipsar dem om denna.

*Hur jag jobbar mycket mer i sådana fall är att jag visar dem någonting och ber dem prova det och sedan säga vad de tycker om det. T.ex. blev jag visad ett Presenter First UI-Pattern, kolla på det här. Sen så testade teamet det och sedan frågar man dem om de vill använda det och då vill jag ha en förklaring till varför, i så fall köper jag det. (Anders Nyberg)*

En fråga som vi ville undersöka var huruvida ScrumMasters tidigare erfarenheter påverkade hans nuvarande roll. Var han van vid att styra traditionella projekt där han organiserade teamets arbete och sa åt dem vad de skulle göra, och påverkar det i så fall teamets ansvarstagande i Scrum? Avdelningens ScrumMaster Lars Ahlkvist har arbetat som utvecklare på SEMC sedan 2003 och han hade tidigare ingen erfarenhet av agila metoder eller Scrum. Han hade tidigare arbetat i projekt som trots att de var baserade på en vattenfallsmodell tillämpade en form av iterativ utveckling. Han hade dock ingen erfarenhet av traditionell projektledning, och kommenterade att han inte såg sin nuvarande roll som projektledare, då han enbart guidade. Ingen av de intervjuade gav oss intrycket av att Lars tidigare erfarenheter skulle påverka hans sätt att sköta rollen som ScrumMaster på ett negativt sätt utan snarare tvärtom, han verkade verkligen anamma Scrum och försökte inte styra teamet på ett kontraproduktivt sätt.

Innan Lars hade en uttalad ScrumMaster-roll hade linjeför Anders Nyberg något av en ScrumMaster roll i projektet, det var på hans initiativ Scrum togs in på avdelningen. Det var på grund av hans tidigare erfarenheter de valde Scrum. De hade märkt att när de flyttade ut till en egen avdelning förlorade de en del av kontakten med dem som använde systemet.

*När vi satt och utvecklade så satt vi nära och fick feedback, och det var väldigt effektivt att köra så. När vi flyttades ut märkte vi att vi förlorade kontakten med folk och då blev det svårt att få reda på vad folk egentligen ville ha. [...](Anders, Nyberg)*

En av ScrumMasters huvuduppgifter är att skydda teamet från yttre störningar under aktuell sprint. Både ScrumMaster och linjechef Anders Nyberg poängterar att de ser nytta med att skydda teamet under sprintarna, och det är något de arbetar efter. I de första sprintarna försökte Product Owner påverka teamet och få dem att ändra saker i sprintarna utan att gå efter Scrumprocessen förklarar Lars Ahlkvist, men nu har de lärt sig hur de ska agera menar de. Lite av problematiken kom på grund av att Product Owner sitter tillsammans med teamet tror Lars, men att fördelarna att ha honom så nära överväger risken för nackdelarna. David Jönsson beskriver att tidigare gick Product Owner mer frekvent direkt till teamet och försökte ändra planeringen under en sprint, men att det har blivit mycket bättre efter de första sprintarna, nu måste Product Owner ta det på ett möte eller gå via ScrumMaster. Att arbetet har förbättrats syns i de Burndown Charts vi har tagit del av, dessa förtäljer dock inte vilka faktorer som orsakat förbättringen.

Det var dock aldrig några större problem kopplat till skyddandet menar David. Karl Eklund förklarar att det var mer problem tidigare i form av en skyddande roll kopplat till ScrumMaster.

*Den stora skillnaden är ju att den första ScrumMastern var ju sektionschefen Anders, och det var ju ett jätteproblem, bland annat för att det ju var mot honom som ScrumMaster skulle skydda oss. Och hur duktig du än är på att ha två hattar så är du fortfarande bara en människa, och hans prioriteringar som sektionschef slog ju igenom hans ledning som ScrumMaster. (Karl Eklund)*

En annan viktig uppgift för en ScrumMaster är att utbilda och guida de inblandade om Scrumprocessen. Varför man gör saker och hur man gör dem. I vårt undersökta projekt har utbildningen av Scrum skett i två omgångar, först av linjechefen Anders Nyberg som i början agerade som en sorts ScrumMaster, och sedan i form av en uppdatering av Lars Ahlkvist i rollen som nyutbildad ScrumMaster. Nyberg tog upp Scrum och dess grundläggande principer och riktlinjer, och Ahlkvist kompletterade med förtydliganden kring de rutiner och förfaranden som genomförts felaktigt eller annorlunda. Karl Eklund och David Jönsson nämner den initiala perioden som problematisk, då kunskapen om Scrum ännu inte förankrats i gruppen:

*Vi hade också väldigt dålig kunskap, kändes det som, i skillnaden mellan Backlog och Scrum-items. Så det var väldigt svårt, tyckte jag, det första Scrumplaneringsmötet jag var med om, att förstå meningen med att sätta timmar på allt var eftersom det inte blev en fast lista ändå. Och det har ju löst sig. (Karl Eklund)*

*Det var en inkörningsperiod för att lära sig hur det fungerade, tog väl en eller två sprintar innan det kanske började fungera. (David Jönsson)*



Förutom dessa informella utbildningstillfällen kommenteras det att gruppmedlemmarna lär sig mer om Scrum efterhand som arbetet fortskrider, tack vare att ScrumMaster hela tiden finns där för att förklara när det finns några otydligheter. Denna kontinuerliga utbildning är en del av att ScrumMaster guidar teamet under sprintarnas gång, och ett litet exempel tas upp av David Jönsson:

*Jag läste en annan artikel som Lars, vår ScrumMaster tipsade mig om [...] (David Jönsson)*

Som vi nämnt tidigare var detta även tydligt vid observationerna av Daily Scrum, då Lars som ScrumMaster hjälpte till genom att guida teamet. Även observationerna av arbetsmiljön stödjer detta argument.

Samtidigt som ScrumMaster har ansvar för att lära ut och förankra Scrum hos teamet, så är det hans uppgift att lära upp Product Owner, då denne har mycket kontakt med teamet, och därför behöver vara insatt i hur arbetet går till. Product Owner ska även sköta sina ansvarsområden inom Scrum, så som att se till att sköta Product Backlog. Då är det viktigt att han har fått en god grund och tagit på sig ansvaret.

*Vi har pratat mycket om det, och det börjar bli bättre, men han måste ha en full Product Backlog, och den måste vara prioriterad. Han kan ju inte lägga till grejer som vi ska titta på. Vi ska faktiskt lägga en viss tid av sprinten på att estimerar Product Backlog och då måste där vara grejer också. (Lars Ahlkvist)*

En ScrumMaster måste även övervaka och reglera Scrumprocessen. Lars Ahlkvist har i sin roll som ScrumMaster god översikt över arbetet som sker inom teamet, tack vare att han sitter så nära dem, och tack vare att han är noga med att hålla i alla möten. Under våra direktobservationer (Både Daily Scrum och arbetsmiljöobservationerna) var det tydligt att han hade denna översikt, vi såg honom exempelvis stå och diskutera en Burndown Chart med en teammedlem.

Han är även snabb på att upptäcka brister och problem, och att agera på dem innan de eskalerar enligt teamet själva. Detta beskrivs väl av Niklas Holmberg, Karl Eklund och Jonas Nyberg:

*Han ser ju till att saker och ting blir gjorda, hos oss är han ofta den personen som många har tilltro till tekniskt, har lösningar, och om han inte har det så frågar han produktägaren direkt: "hur ska vi göra i det här fallet?" (Niklas Holmberg)*

*Han är snabb på att identifiera när folk försöker.. maska är fel ord, men när folk inte klarar av att lösa uppgiften, när det börjar dra ut på tiden, när folk är osäkra, så är han snabb på att påpeka det, och peka ut någon: "du gör inget just nu, går över och prata med honom", i de fallen där folk inte själva säger att de behöver någon. Så vi behöver det nu i början, vi behöver en liten spark, eftersom det är ett ovant sätt att arbeta. (Karl Eklund)*

*Det är mer guidning tycker jag, och sen har han ju koll på utvecklingen i sprinten och fångar upp om vi halkar efter eller ligger före. (Jonas Nyberg)*

Lars sköter alltså sitt ansvarsområde att övervaka och reglera genom att hela tiden se till att medlemmarna i teamet inte stöter på några hinder de inte kan klara av själva, utan att börja lägga sig i och detaljstyra. Han beskriver själv sin roll som följande:

*Huvuduppgifterna är ju att se till att de andra kan utföra sitt jobb. Jag är ju utvecklare dessutom, så jag känner mig delaktig i att se till att allt kommer i mål. (Lars Ahlkvist)*

Allt detta är konsekvent med det som skedde på de två Daily Scrum Meetings som observerades, där Lars stöttade medlemmarna och kom med input där det behövdes. Han visade även att han var insatt i de problem som existerande under den aktuella sprinten, och uppdaterade medlemmarna om dessa.

#### 4.2.3 Product Owner

Flera av informanterna tog upp ett problemområde inom Scrum, att Product Owner måste kunna släppa kontrollen. En vilja att i stor grad påverka och detaljstyra fanns i de första sprintarna, och detta är något som kommenteras ett flertal gånger, bland annat via citat som följande, angående att hålla isär sina roller:

*Nej, jag har inte det, men vår Product Owner var ju project manager innan vi började med Scrum. Jag det ser jag inte som en nackdel, däremot att komma från en bestämmande roll till att gå ner till Product Owner eller ScrumMaster tror jag är svårare. Man vill säga ifrån när man är van vid att säga ifrån. (Lars Ahlkvist)*

Detta stöds av övriga medlemmarna i teamet, exemplifierat som följande av Karl Eklund:

*Vi hade ju inte riktigt det problemet, eftersom för inte alls länge sen hade Product Owner väldigt stort inflytande även under Sprinten. [...] (Karl Eklund)*

Data från flera intervjuer tyder på att detta problem förändrats till det bättre sedan starten, allteftersom teamet fått in vanan att arbeta med Scrum, och allas roller blivit tydligare. Fortsättningen på det senaste citatet lyder såhär:

*Det har blivit lite bättre, men jag tror att om två Sprintar, så kommer det att vara mycket bättre, eftersom Lars är så bra, och är så bra på att säga "jag tycker inte att detta är en bra lösning". (Karl Eklund)*

Återigen visar insamlade Burndown Charts att arbetet förbättrats med tiden, men inte varför. Ytterligare kommentarer kring denna förbättring inkluderar (På frågan om Product Owner stör teamet):

*Nej, det var kanske mer av det första sprinten, som han ofta kom med grejer, men nu är han rätt så inkörd på det här att nu är det sprinten som gäller, och att det är det som man får koncentrera sig på, så det tycker jag inte. (Jonas Nyberg)*

Och följande om hur det är att arbeta nära sin Product Owner:

*Ja ibland har ju kanske Product Owner saker som han vill att vi ska göra som inte finns i sprinten, då ska man kanske försvara sig mot det, i Scrumprocessen. (David Jönsson)*

Så slutligen har vi Niklas Holmbergs (Product Owner) egna åsikter om sin position och sin möjlighet att detaljstyra. Han erkänner själv att det var svårare i början att bryta de gamla mönstren och att viljan att skapa kontroll över situationen var större. Ett kort citat för att belysa detta, angående frustration över att inte kunna detaljstyra:

*I början kände jag det, att jag blivit av med all min makt, och all min påverkan, samtidigt så gör det inte så mycket. (Niklas Holmberg)*

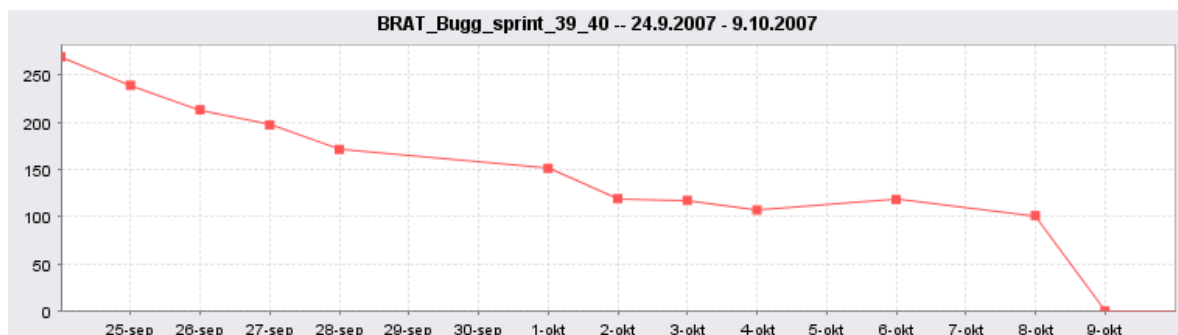
Förutom en ökad vana vid Scrums arbetssätt tillskriver Product Owner sin förbättrade inställning och anpassning till Scrum och tappandet av kontroll till resultaten som teamet nu levererar. Eftersom de levererar enligt schemat och med hög kvalitet så har oron och kontrollbehoven minskat:

*Det fungerar faktiskt mycket bättre än vad jag trodde från början, och det är lite att man är skeptisk som projektledare för produktägare, då man innan hade allt ansvar, att delegera allting och se till att allting blir gjort och man känner att man har hela ansvaret på sina axlar och om jag inte piskade på så skulle ingenting hända. (Niklas Holmberg)*

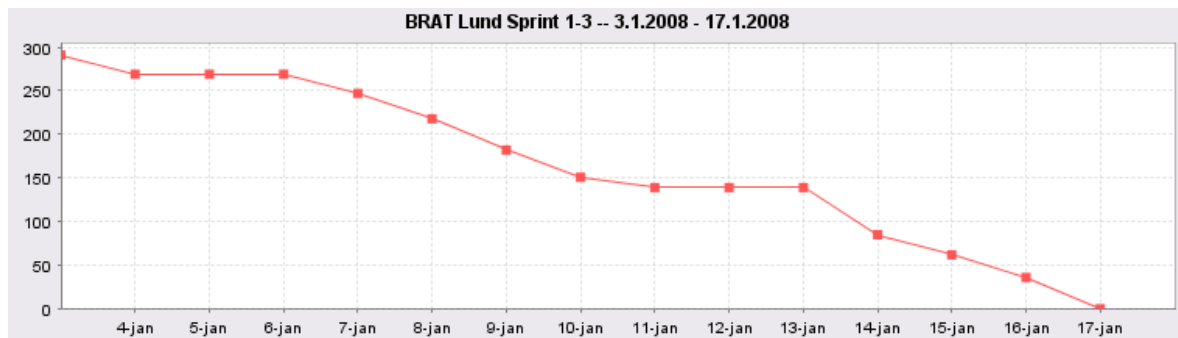
Detta tillsammans med utbildning av ScrumMaster i Scrum har lett till följande:

*Ja, resultaten i sig över vad de lyckats göra, det lugnar ju en. Men i början så tyckte man att det tog för mycket tid, det händer ju ingenting, vi har bara möten, och alla ska vara delaktiga, det kändes så onödigt, jag hade ju kunnat bestämma. Hela det tankesättet förändras ju när man inser att det ju går fortare om alla, inte initialt, men kunskapsnivån höjs och alla får vara delaktiga, de klarar att avgöra saker själva utan att man tvingar dem och säger till vad de ska prioritera. Så resultaten av de första tre sprintarna kanske inte var de bästa, men när vi kom till de senare så tyckte jag att det var väldigt bra, och därför känner jag inga frustrationer längre. (Niklas Holmberg)*

Att resultaten förbättrats syns tydligt av insamlade Burndown Charts (se bilaga 9) och kan exemplifieras med följande två Burndown Charts. Observera att i figur 4.2-1 plockade de bort uppgifter ur sprinten den 8 oktober, detta berodde enligt linjeföraren på att de inte hade kunskap hur de skulle hantera att de inte hann klart med sprintmålen. Notera även att Y-stapeln innehåller ett större tal i figur 4.2-2.



Figur 4.1, Burndown Chart, vecka 39-40, 2007



Figur 4.2, Burndown Chart, vecka 1-3, 2008

En litet fläck i protokollet som dock bör tas upp är att Product Owner ändå indikerar tendenser att ha kvar kontrollbehov. Han erkänner att han hade haft ett större behov av dagliga rapporter ifrån ScrumMaster om han inte hade suttit i samma miljö som teamet, och att han hade haft större krav på det som levererats. Han nämner att han hade behövt bättre definierade processer för informationsflödet mellan honom och ScrumMaster, och att tilliten hade minskat däremellan.

Ytterligare en negativ faktor som tillskrivs Product Owner är att han fortfarande har en tendens att vilja ändra kraven under en sprint, något som ScrumMaster påpekar att han inte får (enligt Scrum). Han påpekar istället att Product Owners ansvar bör ligga (och ligger för nuvarande) på att hålla Product Backlog uppdaterad och fylld med uppgifter som teamet kan arbeta med.

Trots de negativa faktorerna som nämndes i föregående stycke anser teamet att samarbetet med Product Owner fungerar väl. Det som nämndes gång på gång var faktumet att medlemmar i teamet (tack vare att Product Owner sitter så nära, vilket är uppenbart både via intervjuer och direktobservationer) har goda möjligheter att kontinuerligt fråga honom om råd när det gäller saker som Backlog Items eller kravspecifikationer:

*Däremot så har vi kommunikation med användare osv. för att lösa uppgifter, då kan vi gå till Product Owner och fråga "vem är det som bestämmer hur det här ska fungera?", det kan vara en löst specad grej, då säger han att på just den avdelningen fungerar det så här, då kan vi prata med den avdelningen. (Lars Ahlkvist)*

*Ja, han kommer gärna med frågor och sånt, samtidigt som Team-medlemmarna har lättare att bara hugga produktägare och säga typ "kolla här, var det såhär du menade?" (Lars Ahlkvist)*

Både Martin Sternebring och Karl Eklund tar sen upp ett orosmoment i att inte kunna ta del i den övergripande och långt sträckande strategin för projektet, med argumentationen att Product Owner inte alltid är kapabel att tillhandahålla den:

*Ja det är väll helhetsperspektivet, som jag har förstått det ska väll Product Owner informera om agendan lite övergripande, lite mer. Vilket han inte har gjort, vilket gör att man kanske inte vet vart man är. Men samtidigt är det hans uppgift att prioritera det som ska göras i sprinten, och det gör han. Men visst hade det underlättat om han kunde berätta lite mer långsiktigt. Dels kan han också göra fel, och teamet inte göra det som är rätt utifrån den riktiga prioriteringen. Om två månader kanske man ska*

*skriva om hela logiken och så vidare, och det kan jag sakna att inte få med. (Martin Sternebring)*

*Jag tror att vi är på god väg till det, men ibland känns det som att han inte förstår att det är andra saker han behöver göra, han behöver tänka två veckor, fyra veckor, sex veckor i framtiden, hela tiden. Han måste uppdatera Product Backlog, och inte bara ta in nya items, utan ”jaha, nu har situationen förändrats, den här sprint-item löstes på ett visst sätt, då måste jag uppdatera de här Backlog items”. Han har sagt det flera gånger, att han inte kan hålla allting i huvudet, och det är sant, det kan han inte, därför måste han utveckla någon form av rutin, att gå igenom det han har. Och han är inte riktigt där ännu, därför blir det lite osynkat mellan honom och vår ScrumMaster. (Karl Eklund)*

En annan uppgift Product Owner har är att förmedla krav till teamet. Ett av problemen för Product Owner när det gäller förmedling av krav till teamet är att slutkunden i det aktuella fallet inte ser sig själv som en kund, eftersom den befinner sig inom samma företag:

*[...] men det fanns inte några externa krav, man gick inte igenom sina egna miljöer som nu, så kravhanteringen är något vi driver själva mycket mer än vad en egentlig kund borde driva till oss. (Niklas Holmberg)*

Även om kravinsamlingen har lyckats, så uttrycker vissa gruppmedlemmar problem kopplade till den. Ett exempel som tas upp är att Product Backlog borde uppdateras för att passa in med långsiktiga mål, och inte bara reflektera de senaste kraven som dyker upp. Ytterligare ett potentiellt problem är att det ser ut att vara oklarheter gällande kontakten mellan kunden och teamet. Anders Nyberg förklarar att kraven kommer i enlighet med Scrum, det vill säga att de går via Product Owner. Samtidigt verkar det finns ytterligare ingångar för kommunikation och krav till teamets medlemmar:

*[...] och när de ställer mig frågor så vill jag gärna veta inte bara vad vi jobbar med den här två veckorna, utan hur ser det ut om fyra veckor. (Karl Eklund)*

#### 4.2.4 Scrum Team

En annan aspekt av kommunikation mellan teamet och ScrumMaster är den som teamet själva initierar, vilket de gör genom att kontakta ScrumMaster i samband med något problem. På frågan om när de går till ScrumMaster svarade två teammedlemmar följande:

*Så det är ju inte så att det måste vara en, två, tre och sen fyra. Och rätt så fort så inser man att alla säger samma sak, och då tar man det till ScrumMaster, och om man gör det när man är två eller om man gör det när man fyra, det spelar liksom ingen roll. När det kommer konsensus att det är någonting som saknas eller något som är ett problem, då tar man det vidare. (Karl Eklund)*

*Ett typiskt exempel är väl om en användare har en felrapport som inte vi kan återskapa, och man efter kontakt med användare känner att man inte når fram, och man fortfarande inte kan återskapa det, vad ska vi då göra? Då kan han vara läglig att fråga vad han tycker om det där, och om vi ska kolla mer på utvärderingen. Ska vi jobba vidare, eller ska vi släppa det och ta de andra grejerna? (Jonas Nyberg)*

Att medlemmarna i teamet kommer till ScrumMaster med frågor kunde urskönjas under observationerna av arbetsmiljön.

#### 4.2.5 Scrum Team, gruppsykologiska faktorer

Att gruppen samarbetar väl är informanterna överens om, och en av de få negativa kommentarerna handlar om att arbetet minskade tillfälligt när en ny person tillfördes till gruppen, och att det därför tog tid att invänja denne vid det nya arbetssättet som Scrum kräver, och vid den aktuella arbetsplatsen. Samtidigt uppföljs detta med att det goda samarbetet bidrog till att invänjningsperioden inte blev längre:

*Nånting som är bra är samarbetet, att alla kan ställa sig bakom någon som har problem så att vi kommer vidare [...](Lars Ahlkvist)*

Just detta fenomen är ett återkommande element, nämligen att medlemmarna i gruppen kan angripa problem gemensamt. Tack vare den nära kommunikation som Scrum uppmuntrar till har teamet lätt för att bli varse om andras problem, och rycka in vid behov. Att ökningen i kvalitet när det kommer till samarbete tillskrivs Scrum finner vi även hos teammedlemmarna, Jonas Nyberg beskriver det som:

*Jag tycker att det främjar verkligen det här ”hjälpas åt”-konceptet., vilket ju inte betyder att man sitter två stycken bredvid varandra och parprogrammerar, men att modellen uppmuntrar att man bollar idéer, typ ”Vad tycker du om det här felet, jag tycker att man kan lösa det på det här viset, vad tror du om det?”. Så på det viset uppmuntrar det samarbete, absolut. (Jonas Nyberg)*

Under de Daily Scrum Meetings som observerades var det väldigt tydligt att samarbetet var väl utvecklat, och att medlemmarna kom med lösningar eller tips om vem som satt inne med den nödvändiga kunskapen. Observationerna av arbetsmiljön gav samma intryck.

Det påpekas även att detta är något som kommit gradvis i samband med att insikten och förståelsen för Scrum ökat:

*[...] på retrospective tog de upp det, att här hjälpte vi verkligen varandra. Initialt tog det lite längre tid, men i slutändan så gick det fortare för helt plötsligt hade vi en bättre lösning, och det gick fortare att implementera. (Lars Ahlkvist)*

Vårt frågeunderlag behandlade frågor om kommunikation och öppenhet inom gruppen. Hand i hand med samarbetet inom gruppen går kommunikationen, där de olika informanterna tagit upp punkter som kunskapsspridning, informell statusuppdatering, samt snabba och öppna kommunikationsvägar.

Kunskapsspridningen sker genom att de som är bra på vissa områden arbetar för att lära ut detta till de övriga medlemmarna i teamet, för att inte en person ska vara oumbärlig när saker och ting går snett, utan att teamet istället ska ha flera som kan rycka in. Att kunskapsspridning var närvarande syntes under direktobservationerna, då det uppstod problem planerades det in i officiella möten för att tackla dessa och sprida kunskapen om lösningarna. Flertalet gånger såg vi olika teammedlemmar stå och diskutera problem de hade i samband med utvecklingsarbetet.

Ansvaret ligger dock inte enbart på de tidigare nämnda experterna, utan varje medlem är ansvarig för att ta del av kunskaperna hos de övriga:

*Ja man kan alltid fråga, sen så kan man få förklarar hur delar fungerar om man vet någon som jobbat med en viss del. (David Jönsson)*

Spridningen sker även konstant tack vare den öppna kontorsmiljön, och att medlemmarna därför i det dagliga arbetet hör de andras problem och lösningar, det exemplifieras med följande citat:

*Samtidigt är det ju också ganska skönt att kunna höra vad folk säger runt omkring dig, för du har ju informella kommunikationsvägar också, som kan vara svåra att upprätthålla ifall alla människor har sin egen enhet. Som det är nu, och någon kommer och pratar känslomässigt om något, så kan man hålla kvar händerna på tangentbordet, lyfta ögonen och spetsa öronen, så är du med i kommunikationen direkt, det är ingen som behöver sprida det vidare, och det är ingen som behöver ha ansvar för kommunikationen. Det är att välja, antingen är du med i jobbet eller i kommunikationen, direkt. (Karl Eklund)*

Därigenom blir man lätt varse om vem som redan löst vissa problem, och vem man bör fråga när det gäller specialområden. Direktobservationerna stödjer denna kommentar, vi såg exempelvis en medlem sitta med hörlurar för att han förmodligen behövde koncentrera sig. Vi observerade även att två teammedlemmar hade en diskussion, och helt plötsligt nämner Jonas Nyberg en sak för dem som diskuterar, han blir en deltagare i diskussionen. Den informella statusuppdateringen sker på samma sätt, tack vare att man hör dessa konversationer förstår man hur långt vissa delar av arbetet har kommit.

Av detta kan man sammanfatta att de kommunikationsvägar som finns är öppna, en åsikt som stöds av teamet, bland annat via följande kommentarer av Niklas Holmberg och David Jönsson.

*Ja, jag tycker att de flesta har känts tryggare i sin roll, jobbar mycket mer utan att fråga en central person som brukade vara med innan, speciellt frågar man varandra mycket mer inom teamet, öppnare kommunikation. (Niklas Holmberg)*

*[Huruvida kommunikationen är bra och öppen] Ja, det tycker jag. (David Jönsson)*

Vår egen observation av arbetsmiljön stämmer väl överens med påståendet att det fanns öppna kommunikationsvägar.

Ett centralt tema för informanterna är förändringen i det kollektiva ansvarstagandet efter att Scrum införts på arbetsplatsen. Med tiden har teammedlemmarna blivit mycket bättre på att ta ansvar inom områden som tidsplanering, arbetstempo, samt att bli klara i tid. Det nämns att tidsestimeringen blivit bättre sen teammedlemmarna blivit tvungna att ta ansvar för sina egna resultat, och därför sina egna estimeringar. Ett talande exempel på detta kommer från intervjun med teammedlem Martin Sternebring.

*Jag tror det är lite så att om en annan sätter tiderna tar man inte det lika mycket på allvar. På allvar gör man men man kan lätt säga att jag inte satte den tiden där, det visste jag från början [att tiden inte var tillräcklig]. (Martin Sternebring)*

Alla informanterna är överens om att ansvarstagandet har förändrats till det bättre, och tillskriver det till införandet av Scrum på arbetsplatsen. Till exempel Karl Eklund och Anders Nyberg tar även steget och beskriver effekten som en av de största fördelarna med Scrum:

*Den stora fördelen är att folk tar ansvar för det som gruppen gör. Det är nog den absolut största fördelen jag upplevt med Scrum. (Karl Eklund)*

*Största skillnaden är alltså att teamet tar ett helt annat ansvar. (Anders Nyberg)*

Detta ansvar sträcker sig därmed ut över mer än endast den personliga arbetsbördan, och gruppmedlemmarna har större intresse för hur det går för övriga kollegor, och en större motivering till att hjälpa dem när de fastnat med någonting. Denna vilja att hjälpa till var närvarande under direktobservationerna, som vi tidigare nämnt så observerades det bland annat att en teammedlem la sig i en diskussion två andra medlemmar hade för att han kände att han kunde bidra till diskussionen. En faktor inom Scrum som specifikt nämns som påverkande är sprintarnas längd. Att behöva leverera resultat efter bara två veckor sätter mer press på ansvarstagandet:

*[...] när man har en deadline, så har man ambitionen att bli färdiga till dess, om man bara har ett löpande projekt så är det mer "jaja, om detta tar två veckor eller en gör ju inte så mycket". Men när vi nu efter två veckor ska ha någonting att visa.. Just det här att ha något att visa har en stor effekt, man vill ju visa något som är bra. (Lars Ahlqvist)*

En av Scrums produktivitetsförbättringar (Som tagits upp tidigare) ska komma i och med att teamet blir självstyrande och självorganiserande, vilket alla informanter tog upp som något positivt, men att det hade varit något som hade förändrats sedan start. Sedan införandet av Scrum i organisationen har teamet gått mot att bli mer självstyrande och självorganiserande, det görs tydligt att arbetet med att nå denna självständighet kommit långt, då teammedlemmarna har mycket att säga till om vad gäller arbetet, tillvägagångssätt och närliggande faktorer. Det finns till exempel gott om utrymme för medlemmarna att föreslå och driva igenom förändringar både på det övergripande och det tekniska planet, som här på frågan om någon utvecklare kommit med tekniska förslag.

*Ahh det gjorde ju Jonas Granström, Granström heter han. Det gjorde han förra veckan, då tog han upp det här med NDepend, som analyser koden, som man kan köra bygg och så. Han är väl lite drivande med CI och byggservrar och det, har man något så säger man väl till. (Martin Sternebring)*

Gruppen har även rätt att säga ifrån och påverka när förslag om ändrade arbetsförhållanden framförs, och detta ses som en morot, just förmågan att påverka:

*Absolut det gör ju hela tiden att man tänker att man kan förbättra något, hade det inte funnits hade man förmodligen haft tråkigare på jobbet kan jag säga. Absolut. Anders har ju som har ju en dragning någon gång i månaden, varannan vecka vad det kan bli, om en ny teknik som han har hittat. Så får vi bestämma om vi tycker det är bra eller dåligt, så får vi testa det. Har vi hittat en teknik som vi tycker är nice så har vi en dragning, så det är ju öppet, ett öppet klimat får man säga. (Martin Sternebring)*



Ytterligare ett bevis på självorganisering är att gruppen själv hanterar interna roller, så som domänexpertis, och på frågan om hur gruppen sköter detta svarar ScrumMaster:

*Ja, fast det skapar de själv, så det har de löst själva. Det är bättre, med detta kommer mer information till de andra, och de [gruppmedlemmarna] försöker lära dem [varandra] och hjälpa dem. (Lars Ahlqvist)*

Till viss del uppenbarade detta sig även under direktobservationerna, då det var vissa individer som erbjöd hjälp inom olika områden.

Teammedlemmarna nämner att denna hantering av interna roller inte är något som bestämts formellt, ens av dem, utan istället är något som utvecklats omedvetet och kontinuerligt. Det nämns även att domänexpertisen alltmer minskas i takt med att kunskapen sprids till de övriga medlemmarna, och därmed planar ut rollerna och gör dem mer utspädda:

*Ja, viss kunskapsspridning är det ju hela tiden, men det är ju fortfarande så att vissa kan vissa delar bäst. Men jag tror att vi försöker tänka på det, att det inte är en person som ska veta allt när det blir fel. (Jonas Nyberg)*

Det finns dock röster som anser att rollfördelningen inte är så väldefinierad, med exempel från Martin Sternebring och David Jönsson:

*Men några roller tycker jag nog inte. Det är det som är fördelen med dagliga möten, att man vet vad alla jobbar med. (Martin Sternebring)*

*[...] men inte så utmärkande i vårt team, att någon person har specialkunskaper, jag tror vi löser det generellt alla kan lösa alla problem. (David Jönsson)*

Trots alla de positiva kommentarerna finns det tecken på att självständigheten inte har nått ända fram, och att det fortfarande finns ett litet beroende på ScrumMaster, även om han gör sitt bästa för att arbeta bort även det:

*De vill gärna rådfråga mig, men det är de som bestämmer, så det är ju väldigt mycket upp till ScrumMastern, att om han säger något så kommer de nog att lyssna, iallafall här. Men där är det upp till mig att jag måste ta steget tillbaka, "det är inte jag som bestämmer, det är ni som bestämmer själva". (Lars Ahlqvist)*

Denna åsikt återfinns även hos gruppmedlemmarna i varierande grad, med ett exempel i följande svar på vad som skulle hända om ScrumMaster bytte jobb eller slutade:

*Ja det kan ju ha en liten avstannande effekt kan man tänka sig, beror på vem som tar över. Men det känns att han är lite mer drivande än normalt, men det kan ju bero på att han kommer från utvecklarbakgrunden och tycker det är kul och så. (Martin Sternebring)*

Liknande svar fås på frågan om vad som händer när Lars är sjuk.

*Då håller de mötet själva. De organiserar det själva. (Lars Ahlqvist)*

För att slutligen knyta ihop självbestämmandet med ansvarstagandet beskrivet tidigare, så följs kommentarerna om tidsestimering upp med följande svar på frågan om teamet påverkats av att själva fått göra tidsestimeringar.

*Jag tror det kan ha effekten för i början var det jättesnålt med tid, då vågar ingen liksom styra upp någonting. För då känns det som att det inte är min roll att göra klart det på utsatt tid. Det tror jag spelar roll. Så den självförmågan att sätta en tid tror jag ökar, och förra sprinten nådde vi ju ända fram. (Martin Sternebring)*

Vidare behandlade intervjuerna hur sammansättningen av teamet fungerade, och alla informanter är överens om att sammansättningen av teamet är väl utförd. Deras åsikter om denna sammansättning går att dela in i två kategorier, yrkessammansättning och värderingssammansättning.

Att de har liknande åsikter om hur arbetet ska skötas, den yrkesmässiga sammansättningen, beskriver de som följande:

*Ja, rätt så homogen tycker jag, det är inte alla som tänker likadant, men de flesta tänker rätt så likartat när det gäller kodandet. (Jonas Nyberg)*

*[Angående gruppen och hur arbetet ska skötas] Den är ganska homogen ja. (Karl Eklund)*

*Ja det varierar väl, men vi är bra på komma fram till lösningar tillsammans, men rätt lika är vi. (David Jönsson)*

En faktor som tas upp är att alla som är inblandade i projektet har samma inställning till hur själva programutvecklingen ska ske, och att målet är hållbar programkod istället för snabba lösningar som inte är lika flexibla i framtiden. Bland annat Jonas Nyberg och Karl Eklund uppmärksammar detta och anger det som en förutsättning för ett lyckat resultat på projektet.

Teamet är sammansatt så att det ska innehålla de nödvändiga kunskaperna och färdigheterna, och åtminstone David Jönsson uttrycker att han tycker sammansättningen är gjord med det i åtanke:

*Jag tror de teamen vi har nu är väldigt genomtänkta beroende på vad vi har för roller i gruppen, bra sammansättning (David Jönsson)*

Vad gäller de aspekter som gäller värderingar och sammansättningen där så är det återigen samma svar, nämligen att medlemmarna i gruppen fungerar väl tillsammans, och att det därför inte uppstått några större konflikter. Martin Sternebring och Jonas Nyberg beskriver det som följande:

*Värderingar? Ja det tror jag, de jag har hunnit prata och lära känna tycker jag nog. Alla är ganska normala. (Martin Sternebring)*

*[Varför det inte blivit konflikter] Det är en bra fråga, men jag tror det är för att vi tänker någorlunda lika, då är det lättare att man förstår varandra. (Jonas Nyberg)*

#### 4.2.6 Scrum Team, individuella faktorer

En av de individuella faktorer vi ville undersöka var hur en systemutvecklars tidigare erfarenheter påverkar hans förmåga att arbeta efter Scrum och agila tekniker.

När det gäller tidigare erfarenheter i form av hur länge de arbetat som systemutvecklare fanns det stora variationer inom teamet, tre personer hade relativt lång erfarenhet och två personer hade ganska nyligen börjat arbeta som systemutvecklare. När det gäller erfarenhet av någon form av agilt förhållningssätt var den enda personen med reell erfarenhet Martin Sternebring som på en tidigare anställning arbetat efter en iterativ utvecklingsmetod vilket exemplifieras med följande citat.

*Innan Cybercom jobbade jag på ett företag med enormt täta kundkontakter, det var ju väldigt agilt. Det var ju någon sorts projektmodell som jag pratade om, men varje dag så princip ändrades kravspecifikationen så det blev ju agilt och det var lite scrummish. [...] (Martin Sternebring)*

Utöver Martins erfarenheter var det endast Lars Ahlkvist som hade någon form av erfarenhet av agilitet, han hade arbetat med Scrum och agila tekniker så som Test Driven Development (TDD) ungefär i ett halvår. Detta projekt var den första praktiska erfarenheten av ett agilt förhållningssätt för övriga teammedlemmar, även om vissa någon gång hade stött på någon agil teknik, Jonas Nyberg nämnde att de hade använt användarfall i något utvecklingsprojekt han medverkat i. För en sammanställning av teamets tidigare erfarenheter se tabell 4.2.6.1, Tidigare erfarenheter.

När det kommer till erfarenhet av att arbeta med olika systemutvecklingsmetoder så hade alla teammedlemmar arbetat efter någon form av vattenfallsmetod. De kallade metoderna för olika saker men det hade alla det gemensamt att man först gjorde en initial fas där alla krav samlades upp och sedan satte man igång med utvecklingen, dessa projekt styrdes av en traditionell projektledare. David Jönsson beskrev hur han hade arbetat tidigare på följande sätt:

*[...] de körde inte Scrum utan de körde vanlig traditionell utvecklingsmodell där de hade en inpool med fall, buggar och så vidare. Sen hade man ett möte där den gruppen jag ingick i analyserade fallen, en arkitekt och kanske någon utvecklare också. Sen så gick man igenom estimering, och sedan fördelades fallen på utvecklarna. (David Jönsson)*

Huruvida medlemmarnas tidigare erfarenhet av vattenfallsmodellen påverkar deras nuvarande sätt att arbeta existerade det en ganska överensstämmande uppfattning om, att det till största delen var positivt men att det även kunde orsaka problem. Det påpekades av två av de intervjuade att med tiden har man lärt sig att arbeta på ett visst sätt, man för med sig dessa erfarenheter av vana, även om de kanske inte är optimala.

*[...]man har ju med sig sin historik, med åren har man ju arbetat fram ett visst sätt att arbeta, så det har man ju såklart med sig nu även fast man försöker tänka i de här banorna med Scrum, så visst påverkar det. (Jonas Nyberg)*

Han beskrev hur det kan påverka honom på följande sätt:

*Det kan vara att man tänker ungefär att de här kraven är ju inte riktigt utredda, det här borde ju vara bestämt, vara skrivet i eldskrift och hugget i sten. Här är det mer iterativt och man bollar fram och tillbaka. (Jonas Nyberg)*

Martin Sternebring exemplifierade hur det kan påverka som:

*Ja vissa saker är ju ganska uppenbara, skillnaden är att man ska börja tänka att teamet ska göra tidsuppskattningarna istället för en projektledare gör det. Då känner jag att man tappar lite kontrollen på projektet kanske, men samtidigt är det teamet som ska göra det så det har man inte ett team som inte vill jobba så borde det bli lika bra på något sätt om inte bättre då. [...](Martin Sternebring)*

De intervjuade teammedlemmarna ansåg dock att det inte var några stora problem kopplade till deras tidigare erfarenheter. Tre personer beskrev det hela som en process, att det tar tid att komma in i ett nytt sätt att tänka och att de inte riktigt har kommit dit ännu, men att det är något man måste låta ta tid.

*Svårt, det är något som får nötas in, det tänket. Det kanske är lättare för andra än för vissa. Jag tycker man har fått lära sig att tänka om, från hur man gjort tidigare. Man får låta det har sin gång. (David Jönsson)*

*[...] det är en liten tankeändring och det är det jag håller på och bearbetar. [...]  
(Martin Sternebring)*

Sammanfattningsvis kan teamets tidigare erfarenheter visas med följande figur.

Namn	År *	Agila **	Vattenfallsmodell
Lars Ahlkvist	13	Ja	Ja
Jonas Nyberg	11	Nej	Ja
Martin Sternebring	4	Ja	Ja
David Jönsson	1	Nej	Ja
Karl Eklund	1	Nej	Ja

\*Ungefärligt antal års erfarenhet av rollen som systemutvecklare

\*\*Erfarenhet av agila tekniker

**Figur 4.3,** Tidigare erfarenheter

En annan form av erfarenhet många av de intervjuade påpekade var erfarenhet om produkten de utvecklar, och brist på denna produktkunskap kan orsaka problem. Speciellt i samband med tidsestimering. Tidsestimering verkade vara ett problem som blev extra tydligt när någon ny person kom in i projektet, utan kunskap om det som utvecklades. Karl Eklund som var ny i rollen som systemutvecklare och saknade stor erfarenhet av produkten beskrev tidsestimering som någon som han upplevde som fruktansvärt svårt. Detta stöds av följande citat från en av de mer erfarna utvecklarna.

*[...]det vet ju inte en ny person någonting om, eller förstår, det säger dem ingenting. Det kan ta en dag eller fyra veckor, så det problemet kommer vi nog alltid att ha tror jag. (Martin Sternebring)*

Martin Sternebring ansåg dock att estimering även var en erfarenhets- och kunskapsrelaterad fråga, att det är något man blir bättre på ju mer man gör det, att människor som inte har tidsestimerat i stor utsträckning generellt är dåliga på det. Han poängterade att han såg det som ett möjligt problem med Scrum, att man förutsätter att ett team verkligen kan det här med tidsestimering. Han efterlyste en formel för hur man lär ut tidsestimering till personer som kommer in i projektet och saknar kunskap om produkten och/eller estimering.

En viktig aspekt av Scrum är att det krävs kunskap om de tekniker som används och kunskap om Scrumprocessen. När Scrum togs in på avdelningen var det ingen som hade fått formell utbildning inom det. Den utbildning som hade skett leddes av Anders Nyberg och Lars Ahlkvist och de hade i sin tur lärt sig via böcker och Internet, vilket resulterade i en del problem, de visste inte om de gjorde rätt utifrån Scrum men de upplevde det så. Sedan gick Lars Ahlkvist en ScrumMaster utbildning och de insåg att de hade gjort en del felaktigheter men att de var på rätt spår. De största förändringarna som skedde efter utbildningen var enligt Anders Nyberg:

*Sen mycket mycket hårdare på att en sprint ska bli klar, det är inget snack om saken, det är målet som teamet har. Och att vi skulle börja lägga in sådant som helger i sprinten för det gjorde vi inte tidigare utan det är klassificerat på en Burnchart-nivå, man ska lägga in utbildningar, helger, semester osv. (Anders Nyberg)*

Alla de intervjuade upplevde det som att de nu följde Scrumprocessen, de använde olika Burndown Charts, hade sina möten och verkade trivas rätt bra med den nuvarande processen, det var ingen som upplevde det som frustrerande att inte ha full koll på allt i Scrum. Under direktobservationerna agerade deltagarna som om de vore bekväma med arbetssättet. David Jönsson berättar att när han undrar något om processen så frågar han ScrumMaster eller så kollar han själv på nätet, så att det är något som löser sig av sig själv.

Något som har blivit bättre sedan starten är Daily Scrum, i början tog det 35-40 minuter och de intervjuade berättade att det ofta blev diskussioner om hur man skulle lösa ett visst problem. Martin Sternebring menar att det var en kombination av att ScrumMaster inte var tillräckligt hård och att utvecklarna också har lärt sig hur de ska utföra mötet, diskussioner om lösningar tas efteråt med de som vill vara med. Lars Ahlkvist beskrev hela problemet som:

*Det var mer försök till problemlösning. När vi började med Scrum så hade vi morgonmöten på typ 35 minuter eller något sådant, och när vi kommit igång och jag blivit ScrumMaster så var vi nere på en 10 minuter, och nu är vi nere på en kanske 6 minuter (Lars Ahlkvist)*

På båda Daily Scrum Meetings som observerades slutfördes mötena på under 10 minuter, så att de blivit bättre på att hantera Scrumprocessen stöds även där.

Huruvida mer formell utbildning av teamet angående Scrumprocessen hade hjälpt dem att lyckas bättre med införandet av Scrum eller inte fanns det olika åsikter om. I de första

sprintarna var inte processkunskapen så hög, vilket i sin tur enligt de intervjuade resulterade i att sprintarna inte blev speciellt produktiva i form av producerad funktionalitet, men att alla teammedlemmar verkade anse att det är en lärandeprocess, det tar tid och en utbildning hade nog inte hjälpt speciellt mycket. Produktivitetsökningen återspeglas i Burndown Charts.

Sammanfattningsvis beskrivs teamets syn på mer formell utbildning bäst av följande citat:

*Jag tror inte att det är ett måste, det är klart det inte skadar att gå någon lite längre utbildning och lära sig lite mer hur det är tänkt att det ska fungera, men å andra sidan, det beror ju så mycket på vad det är man ska lösa och för system man ska utveckla. Jag tror nästan att det väger in mer att man bollar inom teamet, vad som fungerar bra för oss, "vi löser det på detta viset". Så jag tror ändå att det är viktigare än att man har tung formell utbildning i Scrum. Jag kanske hade sagt något annat om jag gått en bra utbildning, men som jag känner nu ser jag det inte som ett måste att ha gått en utbildning. (Karl Eklund)*

Gällandes den individuella teknikkunskapen och rollfördelning menade David Jönsson att det fanns en viss uppdelning, men mest utifrån vad man gjort tidigare, men att kunskapen om de tekniker de använde var ganska jämn inom teamet. Att kunskapsspridningen om de agila teknikerna de använde var stor höll flera av de intervjuade teammedlemmarna med om.

En fråga vi ville undersöka var huruvida de agila teknikerna kräver mer av utvecklarna i form av erfarenhet, det vill säga den individuella yrkesskickligheten. När det gäller den expertisen kopplad till tekniska lösningar så har teamet hanterat det genom att låta mer erfarna medlemmar ta hand om de mer komplicerade problemen. Detta tycker alla har fungerat bra och inget har motsatt sig detta sätt att arbeta.

Ett under intervjuerna återkommande ämne angående yrkesskicklighet var tidsestimering, och hur de ska förhålla sig till det när det gäller mindre erfarna systemutvecklare, med estimeringen har det uppkommit problem. Något som de flesta påpekade berodde på just erfarenhet och med erfarenhet kommer oftast yrkesskicklighet enligt Martin Sternebring, även om han poängterar det inte är en garant för att kunna estimeras bra. Han menar även att de gånger det kommit in erfarna systemutvecklare i gruppen har deras estimering fungerat bra.

David Jönsson menar att gruppen har blivit bättre på att estimeras, eftersom de har haft bra uppföljning av estimaten och hur de har förhållit sig till de faktiska resultaten, de har fått erfarenhet helt enkelt. David ger även Scrum ett positivt betyg för hur estimering hanteras.

*Att man tar en hel dag för en Sprint Planning Meeting, det har jag inte sett innan, att man tar den tiden. Att tiden det tar att estimeras, och gå igenom verkligen läggs. (David Jönsson)*

Tidsestimeringen under direktobservationerna gick till synes smärtfritt, och skulle kunna vara en effekt till att produktiviteten ökat (som syns i Burndown Charts).

Karl Eklund beskrev estimering som något fruktansvärt svårt, och menade att ju fler sprintar det går, desto ärligare blir man. Detta eftersom om man i början underestimerar en

uppgift får man stå och skämmas och stå till svars för sin dåliga estimering menar han. Vidare menar han att aktuell status för estimeringen är följande:

*Man måste bli väldigt ärlig i tidsestimeringen, och det är den statusen vi är i nu. Sen måste vi dessutom bli bättre på själva tidsestimeringen av nyutveckling och problemkomplexitet, och det är nästa steg. Så jag tror det går väldigt mycket i platåer i utvecklingen i färdigheter. Jag tror det är svårt att både bli ärlig och få in färdigheterna samtidigt, det är väldigt svårt. (Karl Eklund)*

Det faktum att Karl Eklund ännu inte är en fullfjädrad utvecklare som linjeför Anders Nyberg uttrycker det är något som organisationen och gruppen är medveten om, och accepterar, det tar tid att komma in i saker och ting. Karl uttrycker det dock som att han tycker det är ett problem, på frågan om han tycker någonting är frustrerande med Scrum svarar han:

*Ja, att på morgonmötet stå och förklara varför man inte lyckats, det tycker jag är frustrerande, men det är ju eftersom jag är den svaga länken, så är det jag som står där och ber om ursäkt. (Karl Eklund)*

Men att han fick leva med det och att han nog skulle ändra uppfattning ju mer erfaren utvecklare han blev.

*Det är ju inte kul att stå där när man är fullständigt medveten om att man kommer att stå där varje dag och säga "jajajaj, lyckades inte idag heller, jag behöver mer hjälp", det är ju ganska svårt. Jag accepterar det, men det är ju frustrerande. Och det finns ju väldigt få situationer inom mjukvaruindustrin där du är så synlig, som på ett Scrum-morgonmöte. Annars har man ju en deadline som ligger veckor i framtiden, och inte varje morgon. Hade du en dålig dag? Syns direkt på morgonmötet dagen efter. (Karl Eklund)*

Niklas Holmberg (Product Owner) påpekar att det finns en del skillnader i yrkesskicklighet bland utvecklarna, det kan vara verktyg man inte arbetat med, eller att man just inte är van vid estimering för just en viss produkt. ScrumMaster Lars Ahlkvist är också av åsikten att estimeringen blir bättre och bättre, men att det blir en försämring av träffsäkerheten varje gång en ny person som inte arbetat med produkten eller saknar erfarenhet kommer in i teamet. Detta är något som också Martin Sternebring menar är ett problem, som han även hade upplevt på en annan arbetsplats.

*[...]och två som kom i vårt team hade jättesvårt, de kunde inte uppskatta timmar alls. De hade inte jobbat så, så det är jättesvårt för dem. De hade ju ingen aning, så vi fick uppskatta åt dem. (Martin Sternebring)*

#### 4.2.7 Scrumprocessen

Alla intervjuade tog upp att en fördel med Scrumprocessen var att det skapade en form av transparens, att man visste "vem som arbetade med vad, vem som hade arbetat med vad, hur ligger vi till tidsmässigt?", etc. ScrumMaster Lars Ahlkvist poängterade att de var noga med att uppdatera Sprint Backlog i slutet av varje arbetsdag för att säkerställa att den rapporteringen sköttes korrekt, och om de inte skulle göra det skulle de ändå märka det på nästa dags Daily Scrum menade han. Att tidsaspekten i alla fall är med i Sprint Backlog stöds av de insamlade Sprint Backlogs (se bilaga 8, Utdrag av Sprint Backlog).

Lars menade även att det faktum att de kör så korta iterationer (2 veckor sprintar) påverkar transparensen positivt. Eftersom det är så korta cykler får man automatiskt en bättre överblick än om man skulle ha en deadline två månader framåt i tiden förklarade han. Vid direktobservationerna var alla införstådda med målen, då Lars bland annat informerade dem om framgångarna. Jonas Nyberg menade att denna överblick innebar att det var roligare att arbeta än om man skulle arbetat som ensam utvecklare i ett stort projekt med deadline ett år framåt i tiden.

*[...] om man ska jämföra med att vara ensam utvecklare i ett stort projekt som ska vara klart om ett år, då är det ju roligare att det är mer feedback hela tiden på hur man ligger till, i och med att det är så korta cykler. (Jonas Nyberg)*

Jonas tyckte inte att det var påfrestande att inte ha en större överblick framåt i tiden, hade man planerat långt i förväg hade man ändå fått planera om så det hade ändå inte fungerat menade han, det kommer ändringar i alla planeringar. David Jönsson uttryckte sig på följande sätt angående hur Scrum har påverkat hans arbete.

*Om man tar det positiva så får man bättre uppfattning om vart man är i utvecklingsprocessen, hur långt man har hunnit. Det ger en då ju bättre bild om att kommer man att hinna så att säga i teamet. (David Jönsson)*

Niklas Holmberg (Product Owner) gillade också överblicken Scrumprocessen skapade:

*Man har mer koll på var tiden går, det känns som att jag vet vad alla gör, och kan ju hela tiden följa utvecklingen via Burndown-Charts [...] (Niklas Holmberg)*

Martin Sternebring ansåg att det var bra med den överblick två veckors sprintar skapade men att han saknade lite av helhetsbilden, han kände det som om Product Owner inte hade informerat om de övergripande målen tillräckligt, men samtidigt tyckte han att Product Owner skötte sitt jobb eftersom han var noga med att prioritera det som ska göras i sprintarna.

*Men visst hade det underlättat om han kunde berätta lite mer långsiktigt. Dels kan han också göra fel, och teamet inte göra det som är rätt utifrån den riktiga prioriteringen. Om två månader kanske man ska skriva om hela logiken och så vidare, och det kan jag sakna att inte få med. (Martin Sternebring)*

Transparensen var enligt de intervjuade teammedlemmarna orsakad av att man har så mycket kommunikation inom gruppen, dels i det dagliga arbetet skapad av det öppna kontorslandskapet, men även av Scrum i form av att man genomför Daily Scrum varje morgon. Det öppna kontorslandskapet gjorde att man hörde vad andra pratade om, och automatiskt snappade upp delar av samtal som andra personer i teamet hade. Ingen av teammedlemmarna ansåg att det fanns problem att man snappade upp andras diskussioner och dylikt, utan ville man inte lyssna gjorde man inte det. Martin Sternebring ansåg att detta hjälpte gruppen eftersom när man kom till en uppgift hade man förmodligen i bakhuvudet ifall någon i gruppen arbetat med en liknande uppgift tidigare, men han ansåg att detta inte enbart var Scrums förtjänst utan man hade arbetat med denna öppna kommunikation och transparens oavsett projektstyrningsmetodik.

Ovanstående stöds av direktobservationerna då vi såg att de hade två monitorer som visade den aktuella sprintens Burndown Charts, samt statusen över projektets Continuous



Integration. Den sistnämnde monitorn innebar att alla kunde se om någon hade lagt till kod som inte fungerade. Vi såg som tidigare nämnt en stor del av öppen kommunikation om olika frågor gällandes den aktuella sprinten.

Daily Scrum var något alla de intervjuade uppskattade, det hade ju också blivit bättre rent tidsmässigt sedan starten menade David Jönsson, nu tog det 10 minuter att genomföra Daily Scrum. Det enda negativa som uppkom i samband med diskussioner om Daily Scrum var att Karl Eklund ibland kunde tycka det var jobbigt, vilket redan tagits upp tidigare.

Att statusen i aktuell sprint kan övervakas via ett verktyg var något som fem av de intervjuade nämnde som något som verkligen hjälpte dem i arbetet. Verktöget var ett gratis verktyg (ScrumWorks Basic Edition) och linjechef Anders Nyberg hade laddat ner det från Internet. ScrumWorks verkar också ha hjälpt de inblandade att enklare komma in i arbetsgången Scrumprocessen medför, David Jönsson uttryckte sig:

*Vi hade stor hjälp av verktöget Anders hämtade, ScrumWorks som han kanske berättat om. Det tyckte jag var ett bra hjälpmedel det tvingade in en kanske i ett visst sätt att jobba på men just att det gick snabbt att ta åt sig hur det skulle genomföras så att säga. (David Jönsson)*

Utöver att komma in i sättet att arbeta gav det också information om vem som arbetade med vad, teammedlemmarna registrerade vilka uppgifter från Sprint Backlog som de arbetade med, det var även i verktöget de förde in hur många timmar de hade arbetat med en viss uppgift, allt för att skapa så korrekta Burndown Charts som möjligt. Jonas Nyberg förklarade följande:

*Dels så ser man vem som jobbar med vad i verktöget, det är ju rätt så grundläggande. Man ser även förändringen dag för dag hur vi ligger till i tiden. (Jonas Nyberg)*

Detta stöds av de insamlade Sprint Backlog (se bilaga 8, Utdrag av Sprint Backlog), där man tydligt ser vem som arbetar med vilken uppgift, vilken status uppgiften har, och hur mycket tid som det troligtvis återstår. Detta tyder på att de följer Scrumprocessen. Ytterligare en sak som styrker att de följer Scrumprocessen är att man i de insamlade Sprint Backlog tydligt kan se att de har delat upp uppgifter från Product Backlog i flera mindre uppgifter. Detta för att få det hela mer hanterbart som en av informanterna uttryckte sig.

Martin Sternebring menade att man egentligen inte behövde använda ett specifikt verktyg utan det skulle även fungera med exempelvis ett Excel-dokument. Men att ett mer för ändamålet specialiserat verktyg underlättade det dagliga arbetet.

I rummet fanns det en bildskärm som visade tre olika aktuella Burndown Charts hämtade från ScrumWorks. Dessa kunde alla som befann sig i rummet se, vilket även Product Owner påpekade var något som gjorde att han fick bättre insyn gällandes sprintens aktuella status. Dessa kunde identifieras vid direktobservationerna.

Att dessa gruppsykologiska faktorer, till viss del är produkter av Scrumprocessen, berörs mer eller mindre övergripande genom intervjuerna. Informanterna har lätt för att ange Scrum som den orsakande faktorn när det gäller dessa förbättringar, dels för att arbetet med Scrum uppmuntrar och i vissa fall kräver det, dels för att arbetsmiljön främjar det i form av kommunikationsmöjligheter.

David Jönsson nämner just kommunikationen som en förbättrad faktor:

*Men blivit bättre på organisera oss eller vad man säger, förbättrat kommunikationen, ju längre vi arbetat. (David Jönsson)*

Något som vi ville undersöka var om Scrumprocessen skapar vissa av de gruppsykologiska faktorer vi tidigare nämnt. Att teamet har möjlighet att påverka sin arbetsmiljö och sina arbetsrutiner har redan etablerats, Lars Ahlkvist och David Jönsson förklarar att detta sker i samband med Sprint Retrospective Meetings, genom att gruppen tillsammans fattar beslut. Dessa beslut resulterar i en rad mindre ändringar vid varje Retrospective, som på så sätt stegvis ändrar och uppdaterar teamets arbetssätt.

På frågan om vad Lars Ahlkvist hade sagt om utvecklarna kom och förklarade att de inte längre ville köra Scrum på grund av att det inte fungerade, svarar han:

*Då hade jag frågat varför det inte fungerar berätta för mig vad ni vill förändra. Alltså alla förslag vi får från utvecklarna, idéer om arbetssätt så säger jag att vi provar. Sen vid Sprint Retrospective kollar vi på det, var det bra, nej i så fall skiter vi i det. (Lars Ahlkvist)*

Detta tyder ytterligare på att teamet verkligen har makten att ändra det som de tycker är negativt.

På så sätt formar teamet den process de arbetar efter, samtidigt är det dock inte allt som går att ändra lika enkelt, och teamet har förändrat sitt eget arbete minst lika mycket för att anpassa sig själva till Scrum. Återigen är detta en process som sker vid uppföljning av hur väl man lyckats, och att de noggrant går igenom vilka fel och misstag som gjorts, och hur dessa kan korrigeras i framtiden.

Det förekommer även aspekter orsakade av Scrum som inte är lika positiva, men som teamet likväl anpassat sig efter för att allting ska flyta på bättre. Martin Sternebring tar upp ett exempel att man under en sprint bara ska (och får) arbeta med det som är inplanerat under den aktuella sprinten, och att det ibland dyker upp situationer där ett problem måste skjutas upp just för att det inte är inplanerat:

*[...] För att skjuta det till nästa sprint kan kosta 10 gånger mer tid, och det hade jag som projektledare kanske inte köpt. Men man får ju acceptera det, att det är så man jobbar, om resultatet i längden ger. (Martin Sternebring)*

I samma anda påpekar Jonas Nyberg ett annat närliggande problem, nämligen att man i slutet av en sprint nästan hunnit klart med en uppgift, men måste skära ner lösningen för att hinna med deadline, istället för att arbeta en eller två dagar längre och slutföra en mer komplett och långvarig lösning:

*[...] Men det vi stryker kanske inte blir gjort förrän två tre sprintar framåt. Det kan vara frustrerande ibland, men det ingår i modellen att man ska komma till avslut. (Jonas Nyberg)*

#### 4.2.8 Agile och XP

Teamet använder ett antal agila tekniker; Unit Test, Continuous Integration, kollektivt ägande av kod, en form av Test Driven Development, Use Cases samt Code Review. Utöver detta så använder de sig av specificerade Coding Guidelines som utvecklarna måste följa när de programmerar. Unit Test används inte i speciellt stor utsträckning, Anders Nyberg förklarade att det beror på att det är svårt att skriva UnitTest för redan producerad funktionalitet, men när ny funktionalitet skrivs då ska det också ha tillhörande UnitTest. Även Jonas Nyberg påpekade detta problem.

*Vi har börjat införa det, men inte så mycket än. Det är ju så att vi byggde det mesta i systemet innan vi kom in på det här med Scrum och Unit Testing, och det är ju en liten tröskel att införa i ett system som redan är byggt, jämfört med att komma med det från början [...](Anders Nyberg)*

De har inte haft några problem kopplade till införandet av Continuous Integration och kollektivt ägande av kod som kan uppstå då människor inte alltid vill exponera all sin kod för andra. I BRAT-projektet hade dock Continuous Integration inte använts speciellt länge när intervjuerna genomfördes, men de som nämnde det pratade om att de gillade transparensen det medförde. Nu byggdes applikationen en gång per minut och statusen för bygget visades på en skärm som alla kunde se. En av de intervjuade ScrumTeam-medlemmarna gav oss intrycket av att när de väl fått på plats Unit Testing och Continuous Integration kommer det att hjälpa dem med problem kopplade till att tiden i en sprint inte räcker till för att man ska våga göra en sen ändring i någon funktionalitet. Utan en stor del av Unit Test vilket resulterar i en hög så kallad Code Coverage finns inte trygghet för att utföra ändringar eftersom man inte vet om ändringarna man inför orsakar nya buggar, detta blir enligt vissa av de vi intervjuade extra påtagligt i slutet av en sprint.

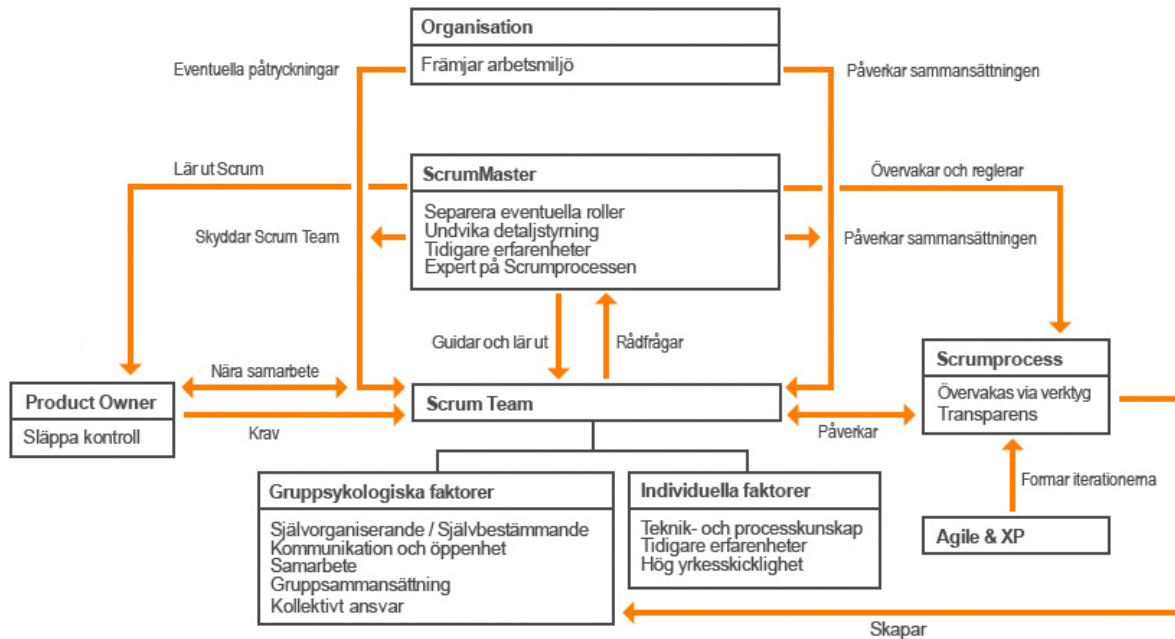
Anders Nyberg berättade att de använde en form av TDD, i den meningen att de skriver Use Cases och testfall innan de börjar koda. Detta var ett arbetssätt de verkade trivas med, och det verkade vara en grundförutsättning för kunna få in mer Unit Tests och ett litet mer klassiskt TDD-tänk. Detta TDD-tänk hittar vi bevis för i de insamlade Sprint Backlogs (se bilaga 8, Utdrag av Sprint Backlog), där vi tydligt ser att många uppgifter är uppdelade i flera olika mindre beståndsdelar; Update Use Case, Write Test Case, Implement samt Test.

Code Reviews används i stor utsträckning och teamet verkar uppleva att det hjälper dem att öka kvaliteten på produkten, och att det är utvecklande rent kunskapsmässigt. Code Review infördes samtidigt som de införde Scrum vilket har resulterat i att de nu genomför Code Review i en större utsträckning än vad som kanske krävs.

*Ja det gör vi, vi Code Reviewar allting, och då gör vi även så att när vi ändrat i en klass så är det inte bara ändringen som ska reviewas, utan då ska hela klassen Code Reviewas. För i början av utvecklingen körde vi inte så mycket Code Review, så nu är tanken att vi ska gå igenom allting successivt. Det är Anders som drivit det här också, det var i samband med att vi införde Scrumprocessen. Vi kände att det var ett bra sätt. Dels att man hjälper varandra, och att det höjer kvaliteten på koden. (Jonas Nyberg)*

### 4.3 Sammanfattning

Resultatet av intervjuerna kan sammanfattas med följande modell:



Figur 4.4, Modell över Scrumprocessen på Test Automation Development

#### 4.3.1 Organisation

SEMC utövar inga direkta påtryckningar på avdelningen gällande sättet de arbetar med undantag för linjeför Anders Nyberg, som också räknas som en del av organisationen. Han gör även det möjligt med en öppen arbetsmiljö med möjligheter till kommunikation och samarbete. Det verkar snarare finnas ett stöd inom SEMC för en arbetsmiljö som avdelningen eftersträvar.

Dock påverkar Anders teamet i form av att han ger dem tips om tekniker han tycker de borde utvärdera, tekniker som han tror kan hjälpa dem i arbetet. Vanligtvis skulle dessa tips kunna ses som direkta påtryckningar, men i detta fallet är det något som teamet inte har något emot, utan ser det positivt att han är drivande i många av dessa frågor.

Det är också Anders som påverkar sammansättningen av teamet, delvis i samarbete med ScrumMaster Lars Ahlqvist. En av orsakerna till Anders påverkan är hans bakgrund som utvecklare och att han tidigare hade en sorts ScrumMaster-roll för teamet.

#### 4.3.2 ScrumMaster

Eftersom ScrumMaster Lars Ahlqvist även har haft rollen som utvecklare har vissa mindre problem och förvirringar uppstått gällandes hans två roller. Teamet har haft svårt att veta

hur och om vad de ska gå till Lars som utvecklare, respektive ScrumMaster. Lars själv menar att ibland har det varit svårt att hålla isär dessa två roller, men att han aktivt arbetat för att göra det. Både teamet och Lars var av åsikten att han inte försöker detaljstyra teamets arbete utan han försöker få dem att ta beslutet själva, vilket teamet uppskattar.

Det finns inget som säger att hans tidigare erfarenheter negativt påverkar sättet han agerar ScrumMaster på, men att hans erfarenhet av vattenfallsmetoder får honom att se styrkorna i Scrum och agila tekniker. Teamet menar att Lars brinner för sin roll som ScrumMaster och att de litar på honom fullt ut i rollen som ScrumMaster. I de första sprintarna fanns det en del problem med sättet de bedrev Scrum på, beroende på att Lars inte hade någon formell ScrumMaster-utbildning. Alla intervjuade menar dock att efter att han genomgått utbildningen har de problemen rättats till. Efter utbildningen ser de honom som en expert, och Lars Ahlkvist verkar också tycka att utbildningen fick honom ytterligare in på rätt spår gällande kunskap om Scrumprocessen. Teamet uppfattar det som att Lars guidar och lär ut dem i frågor om Scrum, en uppfattning som även Lars har. Vissa saknade dock denna guidning i de första sprintarna, men att det nu fungerar tillfredställande. Även Product Owner anser att Lars lär honom hur Scrum fungerar och ska fungera, Product Owner Niklas Holmberg har stort förtroende för Lars kunskaper om Scrum samt för honom som person.

ScrumMaster övervakar och reglerar Scrumprocessen allt eftersom han upptäcker att något inte fungerar, vilket även teamet håller med om. Ett exempel på övervakning är att han medverkar på möten och håller i Daily Scrum. Regleringar skedde exempelvis efter han fått vissa insikter efter ScrumMaster-utbildningen. Han använder även verktygen ScrumWorks för att övervaka status i den aktuella sprinten. Det är förankrat inom avdelningen att teamet är skyddat under iterationerna, i de första sprintarna fanns det problem med att Product Owner störde dem och ville ändra sprintarnas mål. Då fick ScrumMaster gå in och ta tag i problemet och utbilda Product Owner om hur Scrum ska fungera. Detta har de nu tagit itu med och alla informanter tycker att teamet skyddas under en sprint. Krav på förändringar av en sprint måste nu gå genom ScrumMaster, något som gjort att teamet upplever det mindre rörigt, de kan nu koncentrera sig. Det berodde inte enbart på Product Owner utan även på att teamet kanske inte hade tillräckliga kunskaper om sättet Scrum menar man ska arbeta efter, så de satte sig inte emot när Product Owner kom med nya krav.

### *4.3.3 Product Owner*

Under de första sprintarna hade Product Owner Niklas Holmberg svårt att släppa kontrollen, han hade tidigare varit produktansvarig för BRAT vilket både han själv och andra angav som en orsak. Niklas kände i början att han hade blivit av med all sin makt och möjlighet att påverka, något som ändrades när han såg att teamet levererade resultat och att det var bra kvalitet i det de producerade. Det var en inkörningsperiod för Niklas tror ScrumMaster Lars Ahlkvist, något som teamet också håller med om, att det egentligen bara var ett problem de första sprintarna. Dock indikerar Product Owner att han fortfarande inte helt har släppt hela sitt kontrollbehov då han på frågan om han hade agerat annorlunda om han inte delat samma yta som teamet svarar att han då hade krävt in mer rapporter från ScrumMaster, och att hans tillit till teamet troligtvis hade minskat om han satt i en annan byggnad.

Trots de negativa faktorerna som nämndes i föregående stycke anser teamet att samarbetet med Product Owner fungerar väl. Det som nämndes gång på gång var faktumet att medlemmar i teamet har goda möjligheter att kontinuerligt fråga Niklas om råd när det

gäller saker i Backloggen eller djupare information om vissa krav. Detta är möjligt eftersom han sitter i samma rum som Scrum Teamet.

En av Product Owners uppgifter är att förmedla krav till teamet, något som teamet tycker fungerar bra. De upplever dock att de saknar en viss överblicksbild, att Niklas kunde informerat om de långsiktiga målen lite mer. Teamet tycker dock att han nu sköter sina huvuduppgifter på ett tillfredsställande sätt, att prioritera uppgifter i Backloggen och förmedla krav.

#### *4.3.4 Scrumprocessen*

Det finns en tydlig transparens i Scrumprocessen i och med att alla har lätt att se hur läget med projektet är, samt vem som arbetar med vad och vilka problem som lösts. Att be om hjälp är därför enklare, då de på ett bättre sätt kan få reda på vem som har kunskapen de behöver. Denna transparens fås genom den öppna kommunikationen, och även genom att Daily Scrum genomförs varje morgon.

Sprintarna övervakas med hjälp av ett verktyg, ScrumWorks, som medlemmarna är överens om att det hjälper deras arbete med processen, och var ytterligare en faktor till att processens transparens ökade. Tack vare att alla använder samma verktyg för att exempelvis mata in hur mycket som återstår på en uppgift, går det att på ett snabbt och enkelt sätt få fram målade Burndown Charts, ett resultat som uppskattas av de inblandade.

Scrumprocessen är direkt ansvarig för att ha skapat eller förstärkt de gruppsykologiska faktorerna inom teamet, så som ökad kommunikation tack vare öppenhet, ökat samarbete och självorganisering tack vare bristen på detaljstyrning, och kollektivt ansvar som en följd av detta. Den påverkar även arbetet som teamet genomför i den mån att de anpassat sig för att på bästa sätt uppfylla kraven som Scrum lägger fram. Exempel på detta är ändrade arbetsrutiner för att nå deadlines i slutet av sprintar, att fokusera på de inplanerade uppgifterna och inte tänka framåt flera sprintar i förväg, samt att genomföra alla möten och ritualer som tillhör Scrum. Samtidigt så påverkas Scrumprocessen i sin tur av teamet, då de vid varje Sprint Retrospective Meeting går igenom vad som var bra och vad som inte var det, och därefter beslutar om att omstrukturera arbetet och processen för att passa deras ändamål bäst. Makten att göra dessa ändringar har de, och även viljan att utnyttja den.

#### *4.3.5 Scrum Team, gruppsykologiska faktorer*

Teamet har blivit mer självbestämmande och självstyrande sedan Scrum infördes. Alla beslut som rör arbetssätt och metodik görs nu av gruppen som en enhet. Det är även upp till teamet att tilldela eventuella interna roller inom gruppen, och att utbilda varandra då det finns kunskapsskillnader. Denna utbildning sker via ett öppet och direkt kommunikationsflöde, som åstadkommit genom inläring i tänkandet kring Scrum, och den öppna arbetsmiljön. Ytterligare fördelar med detta är att det är lättare att snappa upp när någon har problem och behöver hjälp.

Samarbetet har även det förbättrats i samband med Scrum, och teammedlemmarna har nu lättare att tillsammans arbeta fram lösningar, och de har en ny förmåga att gemensamt ställa sig bakom en individuell utvecklare och hjälpa honom tackla ett svårt problem. En del av detta är ett nytt ansvarstagande, som grundar sig i att gruppen själv fått estimerat tiden på sina uppgifter, och därmed känner att det är deras ansvar att leva upp till sina löften.

Teammedlemmarna visar även upp tydligare tecken på att vara självständiga ScrumMaster, så som att de själva organiserar alla möten och allt arbete då han är frånvarande.

Sammansättningen av teamet kan delas in i två kategorier; yrkessammansättning och värderingssammansättning, och inom båda anser gruppmedlemmarna att det är en god sammansättning. De har samma åsikter om hur arbetet ska skötas och om hur slutprodukten ska se ut, och de har så pass liknande värderingar och synsätt att de inte kommer i konflikter med varandra.

#### *4.3.6 Scrum Team, individuella faktorer*

När det gäller de tidigare erfarenheterna hos teamet så hade alla arbetat efter olika former av vattenfallsmodellbaserade systemutvecklingsmetoder och endast en hade arbetat efter en form av agilt iterativt förhållningssätt innan detta projekt satte igång. Ett problem som identifierades var att planeringen inte alltid skedde som önskat. De menade dock att det inte fanns några stora problem med detta, och alla informanter menade att det var en inlärningsprocess, att lära sig ett nytt sätt att tänka. Det framkom också att kunskap om produkten BRAT var ett problem, att de som inte hade arbetat med den tidigare hade en inlärningsperiod för att lära sig termer etcetera, något som dock borde vara ett problem inom alla utvecklingsmodeller, och inte bara Scrum.

Gällande teknik- och processkunskap så fanns det vissa problem i de första iterationerna i och med att teammedlemmarna inte kände att de hade full koll hur de skulle arbeta i Scrum. De slarvade exempelvis med Daily Scrum, till exempel genom att möten drog ut på tiden för att de inte riktigt visste vad Daily Scrum gick ut på, eller hur det fungerade. Detta var dock något som förändrades till det bättre efter att Lars gick sin ScrumMaster-utbildning.

Alla teammedlemmar tog upp estimering som ett problem kopplat till yrkesskicklighet, att med de som inte hade erfarenhet fanns det stora problem med estimeringen. Något som det även fanns problem med i de första sprintarna. Eftersom resultatet av de första sprintarna inte var tillfredsställande, de hann helt enkelt inte klart, de hade tagit på sig för mycket och estimerat fel. Problemet gällande estimering och mindre erfarna utvecklare såg de ingen lösning på, det var något man fick acceptera. De tar tid att lära sig estimering menade de.

#### *4.3.7 Agile och XP*

Agila tekniker formar iterationerna och teamet använder sig av Unit Test, Continuous Integration, kollektivt ägande av kod, en form av TDD, Use Cases Code Review och Coding Guidelines. Mycket verkade vara relativt nytt för utvecklarna och de hade precis börjat använda Continuous Integration som ett exempel, men de tyckte redan det hjälpte till med transparensen. Unit Test användes bara vid nyutveckling för det var tidskrävande att applicera det på redan producerad funktionalitet. Detta för att programkod måste vara skrivet med en viss abstraktionsnivå för att man ska kunna skapa bra Unit Test.

När det kommer till TDD hade de regler för hur de skulle hantera en uppgift som skulle genomföras, de följer ett mönster där de skriver testfall och Use Cases innan de börjar med själva programmeringen. Vilket kan tolkas som en form av TDD. Något som teamet uppskattade var Code Reviews som de använde ofta, de upplevde att det ökade kvaliteten på produkten samt kunskapen hos dem själva.

## 5 Expertintervju, Jeff Sutherland

---

*I detta kapitel presenteras en sammanfattning av en intervju med en av Scrums uppbovsmän, Jeff Sutherland. Intervjun genomfördes för att möjligheten uppdagade sig, och vi sökte svar på frågor vi funnit i fallstudien, men som vi inte hittat kommentarer till i litteraturen.*

---

En av uppsatsskrivarna fick tillfälle att under 40 minuter samtala med Jeff Sutherland som är en av Scrums grundare. Jeff Sutherland och Ken Schwaber var de som tog fram och formaliserade Scrum från början och Jeff ses som en auktoritet inom Scrum, han arbetar som CTO (Chief Technical Officer) på företaget PatientKeeper i Boston. Utöver det håller han ScrumMaster-utbildningar och säljer konsulttjänster för företag som ska implementera Scrum, exempel på företag han arbetat med är Yahoo, Microsoft, Adobe.

Samtalet behandlade information som hade uppdagats under analysen av fallstudien som vi inte hittade svar på i insamlad litteratur, dessa hade skrivits ner i ett dokument och behandlades i en öppen intervjuform. Intervjun genomfördes under en ScrumMaster-utbildning Jeff Sutherland och Jens Østergaard höll i Köpenhamn den 18-19 januari 2008, intervjuaren medverkade på utbildningen.

Samtalet började med att fallstudien och dess utfall presenterades tillsammans med uppsatsens syfte. Det gick sedan in på frågor angående det faktum att det var avdelningens linjeförman som ofta föreslog vilka agila tekniker som teamet skulle använda i sitt dagliga arbete, vilket enligt vad våra intervjuer antydde berodde på att linjeförmanen hade bakgrund som den person som utvecklade produkten från början. Teamet tyckte att linjeförmans engagemang var positivt, men vi var oroliga för att det i det långa loppet kunde påverka teamets självorganisering och självbestämmande. Jeff berättade att det var vanligt att ett team som kanske inte hade kompetensen eller den tid som krävs för att utforska nya tekniker och hjälpmedel skaffar en sorts teknisk mentor. Han menade att det inte var något stort problem att det var linjeförmanen som gjorde detta, de hade förtroende för honom och det fungerade uppenbarligen för teamet. Vidare berättade Jeff att så länge linjeförmanen hade kvar engagemanget i att driva teamets utveckling när det kom till tekniker och hjälpmedel så skulle det förmodligen inte påverka teamets självorganisering och självbestämmande. Att använda en sorts teknisk mentor som kunde guida teamet till att använda tekniker som skulle öka kvaliteten och produktiviteten var en positiv sak menade Jeff.

Diskussionen gick sedan in på att det krävs stor yrkesskicklighet och erfarenhet för att använda agila tekniker och Scrum. Teamet hade upplevt det mycket svårt för de mindre erfarna utvecklarna att hantera estimering av uppgifter. Jeff berättade att det var något som ofta sågs i team där graden av yrkeserfarenhet varierade, och det var också kopplat till kunskap om produkten, vilket även våra informanter tog upp som en orsak till problemen. Jeff menade att teamet själva skulle hitta ett sätt att hantera det, men att det ibland kunde ta tid, han föreslog Planning Poker som ett bra sätt att lära sig hantera estimeringar. Detta för



att det var ett bra sätt att inte direkt påverkas av andras åsikter när man tidsestimerar. Jeff poängterade också att stor domänkunskap var en av de absolut viktigaste produktivitetfaktorererna hos ett ScrumTeam, hur hanterade teamet kunskapsspridningen av domänen undrade Jeff? Teamet måste ha ett effektivt sätt att informera nytilkomna teammedlemmar om domänen förklarade han.

Ett annat problem avdelningen verkade lida av var att Product Owner fortfarande hade svårt att släppa kontrollen, även om det enligt informanterna hade blivit mycket bättre sedan starten. Jeff fick frågan vad ScrumMaster kan göra med problematik kopplat till Product Owners ovilja till att släppa kontroll. Jeff berättade att i sådana fall var det ScrumMasters skyldighet att försöka få Product Owner att följa Scrumprocessen, detta via samtal och utbildning. Skulle inte det hjälpa var ScrumMaster tvungen att söka stöd högre upp i organisationen och få Product Owner utbytt. Han fick frågan huruvida dessa problem med Product Owner förekom mer frekvent då denne tidigare hade varit någon form av produktchef för produkten som utvecklades. Jeff svarade att det var svårt att säga, det handlar snarare om vilken typ av människa det rörde sig om, och dennes tidigare erfarenheter, inte om han hade varit produktchef eller inte. En produktchef blev ofta en bra Product Owner då denne troligtvis hade stor erfarenhet av att sätta möjligheter och problem i förhållande till reell affärsnytta.

Något informanterna uppgav att de saknade var information om projektets långsiktiga mål. Enligt Jeff var det Product Owners skyldighet att informera om de lite mer långsiktiga och övergripande målen med projektet, annars kunde det skapa minskad trygghet hos ScrumTeamet. I slutändan kunde det även minska teamets tillit till Product Owner påpekade Jeff. Informerade inte Product Owner om dessa mål skulle teamet kräva dem av honom, om han fortfarande inte ville förmedla dem var det upp till ScrumMaster att vidta åtgärder förklarade Jeff det hela.

Under vårt uppsatsarbete hade vi haft svårt att hitta kritik mot agila tekniker och Scrum och ville därför fråga Jeff om lite farhågor vi hade på grund av att Scrum är en kommersiellt paketerad produkt, som utifrån en organisation säljer certifieringar inom Scrum. Under ScrumMaster-utbildningen kallade Jeff Scrum för fail-safe, vilket innebär att Scrum inte kan misslyckas. Har en organisation misslyckats har de inte implementerat Scrum. På grund av detta påstående kom samtalet in på om det fanns en fara i att kalla något för fail-safe. Kan det bli så att många försöker hoppa på tåget och implementera Scrum i en organisation som inte klarar av det, en organisation som inte har den disciplinen agila tekniker kräver? Jeff sa lite skämtsamt:

*“I work with venture capitalists; a bad company deserves to die.” (Jeff Sutherland)*

Samtalet gick sedan in på arvet från vattenfallsmetoder och om vissa fenomen vi hittade i vår fallstudie kunde bero på arvet från dessa? Ett exempel var att teamet satte sådan tillit till verktyget ScrumWorks och att nästan alla informanter nämnde verktyget som något väldigt positivt med Scrum. Detta ter sig lite underligt i förhållande till en av huvudpunkterna i det agila manifestet; Individuals and interactions over processes and tools (se bilaga 2). Jeff berättade att de flesta team inte använder något dedikerat Scrum-verktyg utan sköter det med en Scrum-board eller ett Excel-dokument. En Scrum-board kan exempelvis vara en whiteboard med post-it-lappar som visar hur sprinten fortskrider. Han berättade att Nokia som har över 600 ScrumTeam har förbjudit användandet av ett dedikerat Scrum-verktyg. Jeff trodde att teamet kanske skulle inse att verktyget inte var så viktigt efter hand, och kanske gå ifrån det i ett senare läge. Arvet från vattenfallsmetoder var problematiskt, men

oavsett hur stora problemen var så skulle organisationer ändå gå ifrån vattenfallsmetoder och börja arbeta agilt, systemutvecklingsprojekt kräver ett agilt förhållningssätt och det resulterar enligt Jeff alltid i en bättre produkt. Både från utvecklarnas sida och den som har beställt produkten. Han sa återigen lite skämtsamt:

*”Even the worst Scrum beats a great waterfall project.” (Jeff Sutherland)*

Diskussionen kom sedan in på hur vanligt det var att ScrumMaster även var medlem i teamet, och huruvida det ställde till mer problem än om han inte hade varit det. Enligt Jeff var det inte ovanligt att han var det, men om det fungerade eller inte berodde på organisationen. För vissa team fungerade det, för andra inte som han uttryckte det. Det ställdes givetvis högre krav på ScrumMaster om han skulle kunna hålla isär två roller, och det var teamet som tillsammans med denne fick avgöra om det fungerade eller inte.

Samtalet avslutades med att Jeff pratade om att en av de viktigaste punkterna för ett lyckosamt införande av Scrum är att gruppen har stöd för sitt sätt att arbeta inom organisationen, i denna fallstudie stöd hos linjeföraren Anders Nyberg. Går inte det får man försöka använda en form av gerilla-Scrum, och visa Scrums produktivitetsfördelar med handling förklarade han.

## 6 Diskussion och slutsatser

---

*I detta kapitel presenteras det vi fann i fallstudien i relation till uppsatsens syfte och dess teoretiska bakgrund.*

---

### 6.1 Diskussion

Uppsatsen syfte var att öka förståelsen för hur en systemutvecklargrupps arbete påverkas då de blir delaktiga i en Scrum-implementering med XP-relaterade tekniker, och att utreda vilka faktorer som påverkar resultatet av implementeringen. Vi har valt att lägga upp detta kapitel utifrån dessa två punkter; hur gruppen påverkas, och vilka faktorer det är som påverkar resultatet. Följande diskussion togs fram genom att vi jämförde vår konceptuella modell med den modell vår analys resulterade i.

### 6.2 Hur påverkas systemutvecklargruppens arbete?

#### 6.2.1 Självbestämmande och självorganiserande

Av empirin är det tydligt att informanterna ser sig själva som mer självbestämmande och självorganiserande än de var före införandet av Scrum, samt att de upplever att de nu har goda möjligheter att själva förändra sitt arbetssätt. I de första sprintarna var inte teamet ensamt ansvariga för estimering av uppgifterna, något som de sedan blev. Detta ser vi som ett bevis för att självbestämmande samt självorganisering har ökat sedan starten.

Förmågan att själva påverka sitt arbete är enklare att verifiera, då det är tydligare att medlemmarna i teamet verkligen har denna makt. Vid flera tillfällen nämner de antingen sina valmöjligheter att pådriva förändringar, eller tar upp exempel då detta gjorts med framgång. Vissa av informanterna uppger detta som en morot i sitt arbete, att det driver dem framåt. Detta självbestämmande är även något som Cockburn (2007) och Larman (2004) menar är en av produktivitet fördelarna arbetar då de har ett agilt förhållningssätt, vi har dock inte kunnat bevisa att produktiviteten har ökat i och med detta, men att arbetsmiljön och känslan av delaktighet hos teammedlemmarna ökade. Den teoretiska genomgången gör det tydligt att detta självbestämmande och självorganiserande är centrala delar av arbetet inom Scrum, och att det är viktigt att de uppnås för att införandet av Scrum ska bli lyckosamt.

En annan förklaring till varför deras uppfattning och självbestämmande uppfattas har ökat i och med Scrum skulle kunna vara att linjeföraren inte detaljstyr dem på grund av att inte har tid eller intresse av det. Vi kan inte veta om han hade förhållit sig på ett annorlunda sätt om

avdelningen hade använt en vattenfallsbaserad systemutvecklingsmetod. Men slutsatsen blir att teamet upplever det som en positiv effekt av användandet av Scrum.

### 6.2.2 Kollektivt ansvarstagande

Som den teoretiska bakgrunden visade, är ett viktigt steg på vägen till att bli självbestämmande att lära sig att ta kollektivt ansvar. Teammedlemmarna måste inse och acceptera betydelsen av att nå de för sprinten uppsatta målen, mål som de själva gått med på. Schwaber & Beedle (2002) menar till och med att detta bör vara gruppens huvudfokus, då mycket av arbetet och produktiviteten hänger på det.

När man ser på vår fallstudie ser man att detta stämmer, av den enkla anledningen att informanterna tar upp ansvarstagande som en mycket viktig punkt, samt att de tillskriver den nästan enbart till Scrum. En bidragande faktor till ansvarstagande är även att man inte vill svika teamet genom att inte dra sitt strå till stacken, då det inte är individens prestation som mäts i slutet, utan gruppens totala resultat.

Bland annat tidsestimering tas upp som en tydlig och viktig del av ansvarstagandet. Om man inte själv bestämt vilka tider som gäller, t.ex. deadlines och liknande, är det svårare att känna sig helt motiverad att nå dessa. Om man själv satt upp dem känner man en högre grad av ansvar för att nå den. Det poängteras även gång på gång i litteraturen, det är teamet som är ansvarigt tillsammans, inte en enskild individ. Allt detta borde tyda på att ett höjt kollektivt ansvarstagande är en direkt konsekvens av införandet av Scrum.

### 6.2.3 Kommunikation och öppenhet

Tydlig kommunikation och ett öppet arbetsklimat nämns i empirin som fördelar med Scrum, med motiveringen att Scrum inte bara möjliggör kommunikationen, utan även kräver den. Detta för att kunna skapa den transparens Scrumprocessen kräver, det är också denna transparens som behövs för att nå produktivetsfördelarna som Scrum ska medföra Schwaber (2007).

I litteraturen tas detta upp av bland annat Schwaber (2004), som menar på att om öppen kommunikation inte uppnås kommer de första sprintarna att misslyckas, varpå gruppen tvingas in i ett mer effektivt arbetssätt, som då inkluderar denna egenskap.

Det faktum att teamet hade det svårare under de första sprintarna beror givetvis inte enbart på kommunikationsbrister, men tack vare att förbättringarna inom detta område tas upp som parallella till förbättringarna över lag går det att göra kopplingen, och därmed säga både att kommunikationen förbättrats och blivit öppnare, samt att det kan tillskrivas Scrum. Daily Scrum är ett sådant exempel, det används för att stämma av och informera alla medlemmar i teamet samt ScrumMaster om statusen i den aktuella sprinten. Ingen kan dölja något eftersom om en person försöker dölja någonting uppdagas det fort på dessa möten. Deltagarna i ett Scrum-projekt tvingas på så sätt till kommunikation. Ett annat exempel är att de snabbt märker om kommunikationen med Product Owner är bristande, i vårt fall märkte de detta redan efter två veckor när de tillsammans går igenom vad som genomförts i den senaste sprinten.

I kommunikationen med Product Owner fanns det dock vissa brister, som det att vissa teammedlemmar ansåg sig ha bristande kunskap om projektets långsiktiga och övergripande

mål. Något som enligt litteraturen ligger på Product Owners ansvar, informerar han inte om detta måste ScrumMaster se till att han gör det, vilket ännu inte hade skett i det undersökta fallet. Men när detta sker kommer tillit och kommunikation mellan teamet och Product Owner förmodligen att öka.

#### 6.2.4 Samarbete

Ansvarstagandet, som redan tagits upp som en konsekvens av Scrum, nämns av informanterna som en utlösande faktor till att samarbetet inom gruppen har ökat. Tack vare att alla känner sig investerade i slutresultatet är de också mer benägna att vilja sätta sig in i andras problem och tillsammans komma fram till lösningar. Känslan att varje problem är ett gemensamt problem leder till att samarbetet ökar.

Som intervjuerna och direktobservationerna visat så existerar det ett mycket bra samarbete mellan teammedlemmarna, och att det finns en stark drivkraft i gruppen till att samarbeta.

Litteraturen stödjer vikten av samarbete genom att gång på gång poängtera att ett av syftena med Scrum är att främja samarbete, och att bygga upp en miljö där detta samarbete är enkelt att upprätthålla och utöka. Det är något som vi anser att avdelningen har lyckats med.

#### 6.2.5 Transparens

Flera faktorer i fallet tyder på närvaron av en hög transparens, en faktor som enligt litteraturen är av stor vikt, inte bara i Scrum utan även inom alla agila tillvägagångssätt. Om inte medlemmarna i teamet vet vad den aktuella statusen är, finns risken att de fattar felaktiga beslut och att produktiviteten och kvaliteten därmed blir lidande.

Scrum påverkar transparensen så att den blir mer tydlig, det finns kontrollmekanismer för att försäkra att en hög transparens finns. Daily Scrum är ett exempel, det är väldigt svårt att dölja någonting för de andra i teamet om man varje dag stämmer av hur arbetet har gått.

Intervjuerna, uppbäckade av direktobservationerna, gjorde det tydligt att inblicken i sprintarnas status ökade i takt med att arbetet med Scrum blev mer strukturerat och rutiner angående Burndown Charts och Product Backlog infann sig. Tack vare att dessa Burndown Charts fanns tillgängliga på monitorer, och att de diskuterades under Daily Scrum Meeting, gjorde det lätt att uppnå högre transparens och översikt. Alla teammedlemmar påpekade också att denna transparens var något som verkligen hade kommit i och med att de hade infört Scrum i organisationen.

Informanterna nämner att de verktyg de hade tillgängliga för administration och övervakning av Scrumprocessen ökade dess transparens, de fick tydligare en bild av den aktuella sprintens status. Denna tillit till verktyg var något som förvånade oss. En av det agila manifestets huvudpunkter är:

*”Individuals and interactions over processes and tools” (Agile Manifesto And Principles, se bilaga 2).*

Under intervjun med Jeff Sutherland framkom det att han också blev lite förvånad över detta. Han tyckte det var vanligare att man inte ville ha ett dedikerat verktyg, och att han trodde att de kanske skulle gå ifrån verktyget ju mer de hade kommit in i sättet att arbeta.

Men om det fungerade för dem var det inga problem med att använda ett verktyg. Det kan mycket väl vara så att ScrumTeamet ännu känner sig osäkra på processen. Detta i kombination med arvet från vattenfallsbaserade systemutvecklingsmetoder påverkar dem kanske till att vilja ha ett verktyg för att kunna kontrollera och övervaka processen. Detta via ett bekvämt och vant sätt, nämligen via ett program. De var ju duktiga på att uppdatera informationen i verktyget så det stödde kanske gruppen i transformeringsperioden vi tidigare pratat om.

#### *6.2.6 Agila tekniker som stöd i arbetet*

Tack vare att teamet använder sig av så relativt många agila tekniker (en form av Test Driven Development, Continuous Integration, Code Review, kollektivt ägande av kod, m.m.) så har de en viss effekt på arbetet som helhet.

Informanterna nämner bland annat att Continuous Integration bidragit till en större transparens över lag. Code Review i sin tur sades både öka kvaliteten på slutprodukten och kunskapen hos utvecklarna, och sågs därför som en väldigt användbar teknik. Det är enligt oss även ett bra sätt att få in nya teammedlemmar i sättet de arbetar på.

Ytterligare ett exempel på en teknik som främjar samarbete och kollektivt ägande av kod, och som i andra situationer kan orsaka problem (t.ex. att folk inte vill släppa kontrollen över det de skapat) men som i detta fall fungerar utan problem, i alla fall enligt de teammedlemmar vi intervjuat.

En brist som noterades var att Unit Test ännu inte användes i stor skala, och att ett sådant införande hade resulterat i större trygghet för utvecklarna, eftersom de i dagsläget inte kunde vara helt säkra på om deras ändringar medförde nya buggar eller inte, och detta var ett område där de önskade förbättringar. Detta är något som också litteraturen menar, har man stor del av koden Unit Testad ökar tryggheten och stressen minskar för utvecklarna, de vet att de inte introducerar nya buggar eller förändrar funktionalitet när de vidareutvecklar produkten.

Flera av dessa tekniker infördes i samband med Scrum, och har därför hjälpt till att forma arbetet, och tycks i vissa fall även ha hjälpt Scrumprocessen på traven, t.ex. via ökad transparens.

### **6.3 Vilka faktorer påverkar resultatet?**

#### *6.3.1 Utbildning och processkunskap*

ScrumMaster är den roll i Scrum som ska vara expert på Scrumprocessen, på hans ansvar ligger att utbilda organisationen, teamet och Product Owner angående denna process. För att han ska kunna genomföra denna kunskapsspridning och utbildning måste han således ha full koll på hur Scrumprocessen fungerar. Vår teoretiska bakgrund tog upp att teamet måste ha full förståelse för hur processen fungerar för att de ska kunna se fördelarna med att arbeta efter den, har de inte denna kunskap kan de komma att känna otrygghet och osäkerhet. Dålig processkunskap hos teammedlemmar kan enligt Schwaber (2007) missgynna effektivitetsfördelar såsom transparens och självorganisering, man riskerar då att

oerfarna ScrumMasters detaljstyr teamets arbete. Något som vi dock inte kunde se ha hänt i vår fallstudie.

Våra intervjuer med teammedlemmarna gav dock intrycket av att de i de första sprintarna kände en viss osäkerhet när det kom till Scrumprocessen, för de visste inte riktigt vad den innebar. Ett exempel på där både teammedlemmarna och ScrumMaster visade bristande processkunskap var att under de första sprintarna tog Daily Scrum ofta upp till 40 minuter att genomföra. Möten mynnade ofta ut i diskussioner om hur de skulle lösa vissa problem etcetera, istället för att fokusera på meningen med mötet, att öka transparensen och stämma av arbetet (Schwaber & Beedle, 2002). Detta var något som teammedlemmarna och ScrumMaster upplevde som ett problem, mötet ska gå snabbt menade de.

Vår teoretiska bakgrund uppdagade att denna processkunskap var svårt att få direkt för det var en transformeringsperiod som skedde. Vi fick dock intrycket av att det hade blivit en mindre mödosam transformeringsperiod om teammedlemmarna hade fått mer utbildning om Scrumprocessen redan i ett tidigt skede, före första sprinten, men det hade samtidigt krävt djup kunskap om Scrum hos ScrumMaster redan i ett tidigt skede.

Fallstudien visade på att innan ScrumMaster hade gått sin ScrumMaster-utbildning hade avdelningen missat vissa saker i Scrumprocessen. Den viktigaste saken enligt vår uppfattning var att teamet själva inte var med och uppskattade komplexiteten av Product Backlog, något som förändrades efter genomförd utbildning. Efter det att ScrumMaster hade genomgått utbildningen verkade det som förtroendet för honom ökat, även om det var högt från första början. Detta förhöjda förtroende belystes av alla inblandade personer; linjeföraren, Product Owner samt teammedlemmarna. Ett annat problem vi fann de hade i början var att de behandlade Sprint Backlogen felaktigt i slutet av en sprint. Hann de inte klart med alla uppgifter plockade de bara bort dem ur Sprint Backlog, vilket resulterade i missvisande Burndown Charts. Vilket i sin tur kan skada transparensen (Schwaber 2005), då det ser ut som de hann klart med alla uppgifter vid en snabb kontroll av Burndown Chart. Att hantera detta korrekt ligger på ScrumMasters ansvar. Hade ScrumMaster i ett tidigt skede fått genomgå en mer formell utbildning inom Scrum hade det förmodligen underlättat, han hade kanske även känt sig tryggare i sin roll som ScrumMaster då han hade haft mer kunskap om dess process.

### 6.3.2 ScrumMaster skyddar ScrumTeam

Under litteraturgenomgången fann vi att en av ScrumMasters huvuduppgifter är att skydda teamet från externa påtryckningar för att se till att de får möjlighet att koncentrera sig på att nå de aktuella sprintmålen. I vår fallstudie fann vi att Product Owner under de första sprintarna störde teamet och ville få dem att ändra riktning på sprinten. Detta utan att gå genom ScrumMaster, vilket han enligt Scrumprocessen måste. Detta var ett reellt problem för teamet som de störde sig på och det hade möjligen bakgrund i Product Owners tidigare roll som produktchef för BRAT, han hade helt enkelt svårt att släppa kontrollen. I ett fall som detta är det ScrumMasters skyldighet att se till att Product Owner respekterar Scrumprocessen och att utbilda och upplysa om denna. Detta skedde också, fast det verkar ha tagit några sprintar. En möjlig förklaring skulle kunna vara att det saknades processkunskap hos alla inblandade roller, de visste helt enkelt för lite om processen för att kunna reagera. En annan möjlig förklaring är att en förändring inte sker över en natt, det är en mödosam process att anpassa sig till ett nytt sätt att arbeta, vilket även togs upp i vår teoretiska bakgrund. Att det är en mödosam process som tar tid stöds av de intervjuade

ScrumTeam-medlemmarnas uppfattning om att arbetet med Product Owner har förbättrats avsevärt sedan de första sprintarna.

En annan viktig punkt som uppdagades var att ScrumMaster måste skydda teamet från påtryckningar ifrån organisationen. Fallstudien visade inte på några direkta påtryckningar från organisationen med reservation för att linjechefens påverkan på de agila tekniker teamet använder i vissa fall kanske kan tolkas som påtryckningar, och försök till detaljstyrning. Huruvida detta var ett stort problem fick vi inte insikt om under litteraturgenomgången utan det var först vid expertintervjun med Jeff Sutherland vi fick veta hur Scrum förhåller sig till situationen. Expertintervjun gav svaret att det inte var ovanligt att ett team hade en teknisk mentor som guidade dem i vilka tekniker de kunde testa för att öka produktiviteten och kvaliteten på produkten. Det borde således vara så att om inte teamet själva har kompetensen eller tiden som krävs för att undersöka för dem nya tekniker måste en annan drivande person ta det ansvaret, annars kanske det finns risk att de inte förbättrar sitt sätt att arbeta. Som den teoretiska genomgången tog fasta på handlar Scrum i stor utsträckning om att hela tiden förändra och effektivisera sättet man arbetar på. Inspektera och anpassa (Inspect and Adapt) är en stor del av Scrumprocessens kärna.

### 6.3.3 Sammansättning av ScrumTeam

Litteraturen tar upp att sammansättningen av teamet är mycket viktig, att det ska bestå av medlemmar med liknande värderingar, och att man gärna ska låta ett par initiala medlemmar välja ut resten, för att på det viset öka chanserna för ett team som samarbetar på ett bra sätt.

Verkligheten i vårt fall är istället att teamet är sammansatt av linjechefen med input ifrån ScrumMaster, något som nämns som positivt av medlemmar i teamet. De menar att linjechefen och ScrumMaster har bra koll på gruppens kunskaper, och att de därför är kvalificerade att göra dessa val.

Litteraturen (Coplien och Harrison 2005; Cockburn 2007) menar att likartade värderingar är viktigt för ett teams effektivitet, bland annat personliga värderingar. Våra informanter menade att personliga värderingar inte var speciellt viktiga för arbetet (även om det nämns att medlemmarna har relativt lika sådana), utan att det istället är förhållningssättet till hur de utför sitt arbete som är avgörande. Medlemmarna i teamet har samma synsätt på hur arbetet ska skötas, och tack vare detta kommer de överens. Istället för snabba lösningar förespråkas hållbar utveckling, och detta ses som ett krav för att produktiviteten ska hålla i det långa loppet.

Litteraturen pekade på att liknande personliga värderingar är viktigt för ett teams effektivitet, detta var något vi inte kunde hitta bevis utan det var de arbetsrelaterade värderingarna som spelade roll, något som inte fann i litteraturen. Men samtidigt är skilda personliga värderingar inget som vi kunnat finna påverka effektiviteten negativt. Likheter i de personliga värderingarna kan eventuellt som Cockburn (2007) menar ha minskat konflikterna. Att sammansättningen när det kommer till värderingar är viktig stöds således av fallstudien och den teoretiska bakgrunden, men vår undersökning pekade på att det var de arbetsrelaterade värderingarna som var mest betydelsefulla.

I det undersökta fallet fungerade det att låta linjechefen ansvara för sammansättningen, men det finns inget som säger att det är en garanti för lyckad sammansättning. Det är inte



omöjligt att sammansättningen påverkar mer än vad vi kunnat se i vår fallstudie om exempelvis ansvarig för sammansättningen ej har en lika nära relation till sin avdelning som i vårt fall. Något stöd för Coplien och Harrisons (2005) påstående att teamets effektivitet påverkas negativt om de inte själva får bestämma vilka medarbetare hittade vi således inte.

#### *6.3.4 Organisatoriskt stöd*

Resultatet av fallstudien visade på att det inte fanns några individer utanför avdelningen som lade sig i sättet de arbetar på. De intervjuade teammedlemmarna beskrev att linjechefens stöd angående Scrum var något väldigt positivt, att denne var mycket engagerad när det kom till Scrum och agila tekniker. Således borde man kunna beskriva linjechefens stöd som organisatoriskt stöd. Alla informanter påpekade att stödet och engagemanget från linjechefen var något som de uppskattade och hade stor nytta av. Att detta stöd är ett måste stöds av litteraturgenomgången och speciellt av Schwaber och Beedle (2002) samt Larman (2004). Även expertintervjun med Jeff Sutherland påpekade detta stöd som centralt för införandet av Scrum.

#### *6.3.5 Tidigare erfarenheter och yrkesskicklighet*

Vår fallstudie visade att endast en person i teamet hade erfarenhet av ett agilt förhållningssätt. Alla intervjuade personer var dock vana vid att arbeta efter någon form av vattenfallsbaserad systemutvecklingsmetod. Vår teoretiska bakgrund tog upp arvet från vattenfallsbaserade systemutvecklingsmetoder som ett av de största hindren för ett lyckosamt införande av Scrum. Att detta var ett hinder är inget som stöds av vår fallstudie, en möjlig förklaring kan vara det att alla teammedlemmarna verkade mycket positiva till det agila förhållningssättet. Enda hindret de uppgav var att ibland saknade de en övergripande och långsiktig planering, men de var av uppfattningen att det inte var ett stort problem för teamet. De flesta uttryckte det som att det var något de fick jobba med allt eftersom, att det var en omställning till ett nytt sätt att tänka, en omställning som fick ta tid. Något som även Cockburn (2007) och Larman (2004) påpekar.

Vi fann under vår studie inget av den skepticism gentemot agila tekniker som Cohn och Ford (2003) pratar om utan det bemöttes vad vi kunde se enbart med optimism. En möjlig förklaring till varför vi inte hittade något motstånd mot agilitet kan vara att medeltalet för antal år arbetslivserfarenhet (6 år) inte var extremt högt. Schwaber (2004) och Larman (2004) menar att ju längre en person har arbetet efter ett visst sätt, ju svårare blir det att anta ett annat förhållningssätt.

Något som litteraturen uppmärksammade var att det krävdes hög yrkesskicklighet och reell erfarenhet av agilitet för att kunna arbeta effektivt. Detta fann vi inget stöd av i vår studie då de ansåg sig arbeta effektivt. En alternativ förklaring till varför vi inte fann detta skulle kunna vara att avdelningen ännu inte använder vissa tekniker fullt ut, som exempelvis Unit Test och strikt TDD.

Alla intervjuade teammedlemmar uppgav att det var problem kopplat till tidsestimering av uppgifter, speciellt när det var mindre erfarna med i teamet. Det som fallstudien, den teoretiska bakgrunden och expertintervjun visade var att tidsestimering även är en fråga om domänkunskap, kunskap om produkten de utvecklar.

Detta är således inget problem direkt kopplat till Scrum, utan det är ett problem som troligtvis finns i alla systemutvecklingsprojekt. Det ska poängteras att i Scrum tar men verkligen sig tid att genomföra tidsestimering under ceremonin Sprint Planning Meeting, och alla teammedlemmar är delaktiga i framtagandet av dessa estimat. På det viset blir problemen i samband med tidsestimering kanske mer synliga, i och med Scrumprocessens karaktär. I ett traditionellt projekt är det inte ovanligt att tidsestimeringar görs av en teknisk projektledare, utan att blanda in alla teammedlemmar. Även en av teammedlemmarna påpekade att detta gemensamma framtagande av tidsestimering var något han inte hade sett innan han blev delaktig i ett Scrum-projekt.

## 6.4 Slutsatser

Vår forskningsfråga var följande;

*Vilken påverkan har en Scrum-implementering med XP-relaterade tekniker på en systemutvecklargrupps arbete, och vilka faktorer påverkar resultatet av implementeringen?*

När det kommer till hur systemutvecklargruppens arbete påverkas fann vi att teamets självbestämmande och självorganisering ansågs vara ett resultat av att de använde sig av Scrum. Teamet upplevde att kommunikationen och öppenheten ökat på grund av Scrum, bland annat via Daily Scrum Meeting och den öppna arbetsmiljön. Vi fann att Scrum påverkade det kollektiva ansvarstagandet, det var i vårt fall teamet som tillsammans var ansvarigt. Ett exempel på det är att i Scrum estimerar och planerar teamet själva sitt arbete, teamet sätter upp mål som de tar ansvar för. Detta ansvarstagande har enligt fallstudien även lett till att det undersökta teamets samarbete ökat och förbättrats, exempelvis anses alla problem att vara teamets gemensamma. Vi fann även att inblicken över projektets status ökade markant tack vare användandet av Scrum. Via användande av Burndown Charts, Daily Scrum Meeting etc. fick alla delaktiga en större insyn, transparensen ökade, alla viste vad som skedde och vad status var. När det kommer till hur de agila teknikerna påverkade arbetet fann vi att det formade arbetet och skapade mer transparens och ökad kunskapsspridning.

Vi fann att utbildning och kunskap om Scrumprocessen är en central del för en lyckad implementering av Scrum, och att det måste börja med att ScrumMaster har fullständig koll på Scrum. Även om det är en transformeringsperiod för ett team och dess organisation fann vi att denna hade gått smidigare om ScrumMaster hade fått en utbildning inom Scrum, eller på annat sätt tillskansat sig denna kunskap. ScrumMaster måste sedan sprida denna kunskap till projektets deltagare. Vi fann att denna kunskapsspridning är central för att teamet ska kunna arbeta effektivt och känna trygghet. ScrumMaster måste även skydda teamet mot påtryckning från utomstående. I vår studie fann vi att när det kommer till teamets sammansättning var det av vikt att alla teammedlemmar hade samma förhållningssätt till hur de skulle utföra arbetet, vi fann inte stöd för att de personliga värderingarna spelade in på effektiviteten, men inte heller något som visade på motsatsen. Vi fann även att organisatoriskt stöd upplevdes som något mycket viktigt, utan den skulle de inte kunna bedriva Scrum. När det kommer till teamets tidigare erfarenheter och yrkesskicklighet fann vi inget som tydde på att låg erfarenhet av att arbeta agilt hos teammedlemmarna skulle påverka arbetet negativt. Problem kopplade till yrkesskicklighet uppenbarade sig främst gällandes yrkeserfarenhet av estimering av uppgifter, men där

spelade även domänkunskap stor roll. Scrum gör dock denna typ av problem mer synliga då teamet som exempel är tvungna att tidsestimera tillsammans.

Vi fann att alla teammedlemmar var väldigt positiva till agilitet, även om vissa upplevde att de ibland saknade övergripande och långsiktig planering. Vi fann inget som helst motstånd mot att arbeta agilt i någon av intervjuerna, vilket enligt oss påverkar resultatet av implementationen positivt.

Med denna genomgång anser vi oss att vi har nått syftet med fallstudien och svarat på vår forskningsfråga, men mer forskning inom området behövs.

## **6.5 Vidare forskning**

Ett förslag på vidare forskning är att replikera vår fallstudie hos organisationer som befinner sig i samma skede i en Scrum-implementering som avdelningen vi har undersökt. Detta för att få ytterligare perspektiv på undersökningsfrågan och få möjlighet till att bekräfta eller falsifiera våra slutsatser. De framtida undersökningarna kunde även få möjlighet att ytterligare undersöka teamets självorganisation och självbestämmande, genom att observera de Scrum-ceremonier som vi inte fick med i vår fallstudie. Det hade också varit av intresse att se den replikeras på en organisation där linjeföraren inte är lika drivande, och inte har en så nära relation som linjeföraren har till sin avdelning och sitt team.

## Referenslista

Abrahamsson, P., Salo, O., Ronkainen, J. & Warsta, J., 2002. *Agile software development methods, review and analysis*. Espoo, Finland: Technical Research Centre of Finland, VTT Publications.

Avison, D. & Fitzgerald, G., 2003. *Information Systems Development: methodologies, techniques and tools*. 3rd ed. Berkshire: McGraw-Hill Education.

Beck, K., 2000. *Extreme programming explained: embrace change*. Boston: Addison Wesley.

Beck, K., 2002. *Test driven development: by example*. Boston: Addison-Wesley Longman.

Brittain White, K. & Leifer, R., 1986. Informations systems development success: perspectives from project team participants. *MIS Quarterly*, 10(3), s.215-223.

Bryman, A., 1997. *Kvantitet och kvalitet i samhällsvetenskaplig forskning*. Lund: Studentlitteratur.

Bryman, A., 2002. *Samhällsvetenskapliga metoder*. Malmö: Liber AB.

Ceschi, M., Sillitti, A., Succi, G. & De Panfilis, S. 2005. Project management in plan-based and agile companies. *IEEE Software*, 22(3), s.21-27.

Cockburn, A., 2007. *Agile software development: the cooperative game*. 2nd ed. Upper saddle river, New Jersey: Addison-Wesley.

Cohen, S.G. & Bailey, D.E., 1997. What makes teams work: group effectiveness research from the shop floor to the executive suite. *Journal of Management*, 23(3), s.239-290.

Cohn, M. & Ford, D., 2003. Introducing an agile process to an organisation. *IEEE Computer*, 36(6), s.74-78.

Cohn, M., 2004. *User Stories Applied: for agile software development*. Boston: Addison-Wesley.

Coplien, J.O. & Harrison, N.B., 2005. *Organizational patterns of agile software development*. Upper Saddle River, New Jersey: Pearson Prentice Hall

Creswell, J.W., 2007. *Qualitative inquiry & research design: choosing among five approaches*. 2nd ed. Thousand Oaks, California: Sage Publications.

De Dreu, C., Harinck, F. & Van Vianen, A., 1999. Conflict and performance in groups and organizations. Återfinns i Cooper, C. & Robertson, I., ed. *Organizational psychology and development: a reader for students and practitioners*. Chichester: John Wiley & Sons. Ch 6

De Dreu, C. & Van Vianen, A., 2001. Managing relationship conflict and the effectiveness of organizational teams. *Journal of Organizational Behaviour*, 22(3), s.309-328.

Ehrlinger, J., Gilovich, T. & Ross, L., 2005. Peering into the bias blind spot: people's assessments of bias in themselves and others. *Personality and Social Psychology Bulletin*, 31(5), s.680-692.

Flyvbjerg, B., 2006. Five Misunderstandings About Case-Study Research. *Qualitative Inquiry*, 12(2), s.219-245

Hammersley, M. & Gomm, R., 1997. Bias in Social Research. *Sociological Research Online*, [Online]. 2(1),  
Tillgänglig via: <http://www.socresonline.org.uk/2/1/2.html> [hämtad 30 April 2008].

Higgs, M., Plewnia, U. & Ploch, J., 2005. Influence of team composition and task complexity on team performance. *Team Performance Management: An International Journal*, 11(7/8), s.227-250.

Highsmith, J. & Cockburn, A., 2001. Agile software development: the business of innovation. *Computer*, 34(9), s.120-122.

Hirsch, M., 2002. Making RUP Agile. *Conference on Object Oriented Programming Systems Languages and Applications 02*. Seattle, Washington 4-8 November 2002. Assn for Computing Machinery: Seattle.

Holme, I.M. & Solvang, B.K., 1997. *Forskningsmetodik – om kvalitativa och kvantitativa metoder*. Lund: Studentlitteratur.

Israel, M. & Hay, I., 2006. *Research ethics for social scientists: between ethical conduct and regulatory compliance*. London: Sage Publications.

Judy, K. & Krumsins-Beens, I., 2008. Great Scrums need great product owners: unbounded collaboration and collective product ownership. *Proceedings of the 41st Hawaii International Conference on System Sciences 2008*. Waikoloa, Hawaii 7-10 Januari 2008. IEEE Computer Society: Washington DC, USA

Karakowsky, L., McBey, K. & Chuang, Y., 2004. Perceptions of team performance: the impact of group composition and task-based cues. *Journal of Managerial Psychology*, 19(5), s.506-525.

Kvale, S., 1996. *Den kvalitativa forskningsintervjun*. Lund: Studentlitteratur.

Larman, C., 2004. *Agile & iterative development: a manager's guide*. Boston, Massachusetts: Addison-Wesley.

Law, A. & Charron, R., 2005. Effects of agile practices on social factors. *Proceedings of the 2005 workshop on Human and social factors of software engineering*. St. Louis, Missouri 2005. ACM Press: New York.

- Moore, R., Reff, K., Graham, J. & Hackerson, B., 2007. Scrum at a Fortune 500 Manufacturing Company. *Proceedings of the AGILE 2007*. Aalborg, Danmark 13-17 Augusti 2007. IEEE Computer Society: Washington DC, USA.
- Natale, S., Libertella, A. & Rothchild, B., 1995. Team performance management. *Team Performance Management: An International Journal*, 1(2), s.5-13.
- Norris, N., 1997. Error, bias and validity in qualitative research. *Educational Action Research*, 5(1), s.172-176.
- Mello, A. & Ruckes, M., 2006. Team composition. *Journal of Business*, 79(3), s.1019-1039.
- Patel, R. & Davidson, B., 1994. *Forskningsmetodikens grunder – att planera, genomföra och rapportera en undersökning*. Lund: Studentlitteratur.
- Peslak, A.R., 2006. Emotions and team projects and processes. *Team Performance Management: An International Journal*, 11(7/8), s.251-262.
- Postmes, T., Spears, R. & Cihangir, S., 2001. Quality of decision making and group norms. *Journal of Personality and Social Psychology*, 80(6), s.918-930.
- Rising, L. & Janoff, N., 2000. The Scrum software development process for small teams. *IEEE Computer*, 17(4), s.26-32.
- Schwaber, K., 1995. Scrum development process. *Proceedings of the 10th Annual ACM Conference on Object Oriented Programming Systems, Languages, and Applications*. Austin, Texas 15-19 Oktober 1995. ACM Press: New York.
- Schwaber, K. & Beedle, M., 2002. *Agile software development with Scrum*. New Jersey: Prentice Hall.
- Schwaber, K., 2004. *Agile project management with Scrum*. Redmonton, Washington: Microsoft Press.
- Schwaber, K., 2007. *The enterprise and Scrum*. Redmond, Washington: Microsoft Press.
- Schwaber, K., 2008. *What is Scrum? (About Scrum)* [Online]. Tillgänglig via: <http://www.controlchaos.com/about/> [lästes 27 April 2008]
- Seale, C., 1999. *The quality of qualitative research*, London: Sage Publications.
- Sieber, J.E., 2001. Protecting research subjects, employees and researchers: implications for software engineering. *Empirical Software Engineering*, 6(4), s.329-341.
- Singer, J. & Vinson, N.G. 2002. Ethical issues in empirical studies of software engineering. *IEEE Transactions on Software Engineering*, 28(12), s.1171-1180.
- Stephens, M. & Rosenberg, D., 2003. *Extreme programming refactored: the case against XP*. New York: A! Press.

Szalvay, V., 2004. *An introduction to Agile software development*. Bellevue, WA: Danube Technologies.

Reel, J., 1999. Critical success factors in software projects. *IEEE Software*, 16(3), s.18-23.

West, M., Borrill, C. & Unsworth, K., 2001. Team effectiveness in organizations. Återfinns i Cooper, C. & Robertson, I., ed. *Organizational psychology and development: a reader for students and practitioners*. Chichester: John Wiley & Sons. Ch 5

Wilkinson, I. & Fung, I., 2002. Small-group composition and peer effects. *International Journal of Educational Research*, 37, s.425-447.

Witte, E. & Davis, J., 1996. *Understanding group behavior: small group processes and interpersonal relations*. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Yin, Y., 2002. *Case study research: design and methods*. 3rd ed. Thousand Oaks, CA: Sage Publications.

# Bilagor

---

*Följande material består av de bilagor som är kopplade till uppsatsen, både i form av empirisk data såväl som dokument som använts i samband med insamlandet av dessa data.*

---

## Bilaga 1, Fallstudieprotokoll

### *Introduktion*

Forskningsfrågan denna fallstudie arbetar efter är följande; Vilken påverkan har en Scrum-implementering med XP-relaterade tekniker på en systemutvecklargrups arbete, och vilka faktorer påverkar resultatet av implementeringen?

Syftet fallstudien har är att försöka öka förståelsen för hur en systemutvecklargrups arbete påverkas då de blir delaktiga i en Scrum-implementering med XP-relaterade tekniker, samt utreda vilka faktorer som påverkar resultatet av implementeringen. Detta för att få en större inblick i de problem och faror som finns vid denna process, och på så vis ge ökad förståelse inom området. Nedan visas och förklaras den konceptuella modell som legat till grund för frågeunderlaget.

När det undersöks hur systemutvecklargruppens arbete påverkas undersöks de gruppsykologiska faktorerna som påverkar gruppen samt de eventuella förändringarna i arbetssättet Scrum medför. Med arbetssätt menas hur systemutvecklargruppen utför det dagliga arbetet för att ta fram sin produkt.

### *Fältinformation*

Företag:

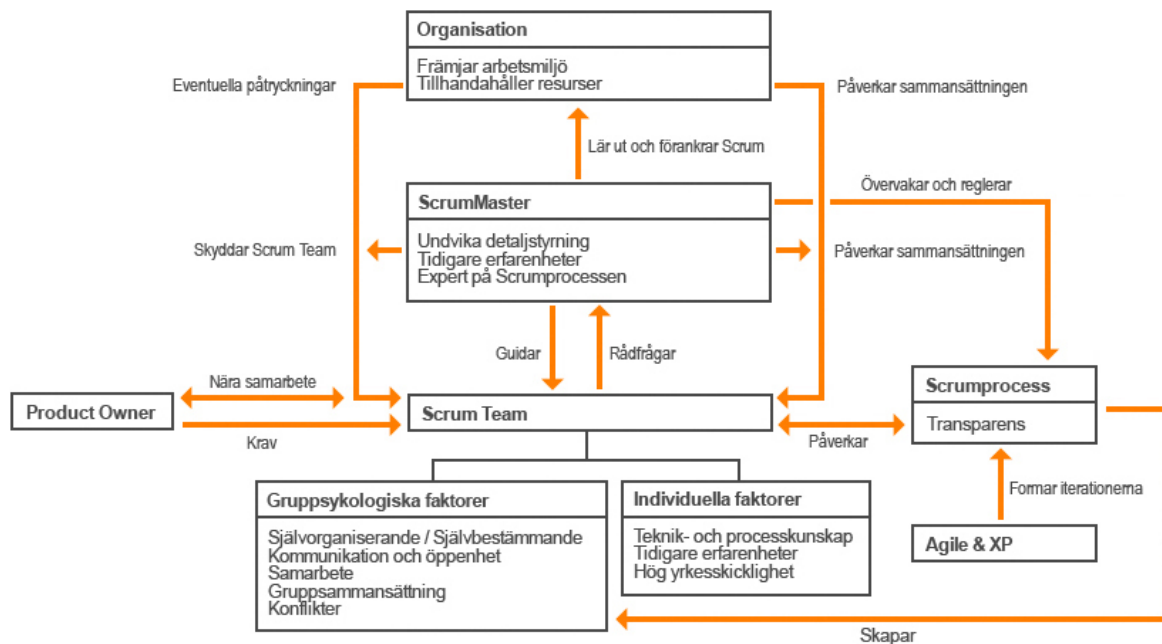
Datum:

Fältarbetare inkl. kontaktuppgifter:

Projektet som ska undersökas ska vara ett projekt som nyligen börjat använda Scrum. Organisationen ska tillhandahålla möjlighet att intervjua alla typer av roller i ett Scrum-projekt; ScrumMaster, Scrum Team samt Product Owner. Fältarbetarna ska också ha tillåtelse att observera Daily Scrum och ta del av eventuella artefakter som exempelvis Sprint Backlog. Åtkomst till undersökningsplatsen ordnades genom att en förfrågan skickades till denna, där syftet och kraven för undersökningen specificerades och utifrån det valde de att delta i fallstudien.



## Konceptuell modell



### Faktorerna som påverkar Scrum Team

En rad olika faktorer påverkar det dagliga arbetet för teamet. Dessa kan i sin tur kategoriseras som antingen individuella faktorer eller gruppsykologiska faktorer.

De individuella faktorerna inkluderar fenomen som kunskap om tekniken och processen, tidigare erfarenheter, samt yrkesskicklighet. Dessa har en direkt påverkan på medlemmarna i teamet då de är en central del av det dagliga individuella arbetet. Yrkesskickligheten krävs för att teamet ska kunna arbeta självständigt och kunna ta egna beslut, mindre erfarna systemutvecklare tenderar att förlita sig på andra för att driva arbetet framåt, på grund av osäkerhet etcetera (Schwaber 2004). Yrkesskickligheten innefattar således även stor kunskap om det som ska utvecklas, så kallad domänkunskap. Cockburn (2007) menar även att en förutsättning för att lyckas med agilitet är att använda duktiga och erfarna utvecklare, då systemutvecklargruppen har relativt få medlemmar krävs det att alla kan bidra. Tidigare erfarenheter är något som kan bidra positivt och negativt. Positiv påverkan handlar om att en utvecklare för med sig kunskap som han kan använda för att öka teamets effektivitet, det kan handla om att han är expert på något område, kunskap som han sedan sprider inom gruppen. Negativ påverkan kan innefatta det som Schwaber (2007) kallar för muskelminne hos systemutvecklare, som exempelvis att man för in ett sekventiellt vattenfallstänk i en grupp som ska arbeta iterativt och inkrementellt, vilket leder till att man motverkar effektivitetsförbättringarna det agila tänkandet ska medföra. Om en individ saknar kunskap om Scrumprocessen, kommer inte personen att känna trygghet i sättet att arbeta. På det viset missgynnas effektivitetsfördelar såsom transparens och självorganisering som Scrum ska leda till, och på detta vis haltar Scrumprocessen som helhet (Schwaber 2007). Detta i sin tur kan leda till att ScrumMaster försöker gå in och detaljstyra teamets arbete, Schwaber (2007) poängterar att detta kan hända då mindre erfarna ScrumMasters ska guida team som har problem med att anpassa sig till självorganisation och självbestämmande.

De grupprelaterade faktorerna, med exempel som samarbete, kommunikation och öppenhet, självorganisation och självbestämmande är skapade av Scrumprocessen. Som ett exempel ökar tilliten mellan medlemmar i gruppen på grund av transparensen Daily Scrum medför, samt den öppna arbetsmiljön som teamet ska verka i (Schwaber 2004). Scrum förutsätter och förlitar sig på dessa faktorer för att nå de effektivitetsfördelar som dessa faktorer bidrar med, fördelar som ökad produktivitet, bättre arbetsmiljö, bättre slutprodukt etcetera. Självorganisation och självbestämmande är det som Schwaber (2007) och Cockburn (2007) poängterar är mest mödosamt att nå, det är ofta en kostsam process att nå dit. Detta på grund av att det är teamet själva som måste göra denna resa och upptäcka fördelarna detta bidrar med, det brukar ta ett par sprintar att nå dit (Schwaber 2004). Detta leder också till att de första sprintarna inte brukar vara speciellt produktiva, gruppens energi går åt till att transformera sig till att bli självbestämmande och självorganiserande, man gör misstag som man sedan lär

sig av och förändrar sitt arbetssätt (Cockburn 2007). Scrums empiriska del träder således in (Schwaber 2004). I och med denna mödosamma resa riskerar man också att ScrumMaster försöker kontrollera och beordra teamet, vilket går helt emot dennes uppgift och arbetsroll (Schwaber 2007). Det kan handla om att teamet själva måste lära sig att lösa konflikter inom gruppen, att själva lösa de problem som uppstår i det dagliga och inte förlita sig på att någon annan, som exempelvis ScrumMaster ska lösa dessa problem åt dem. När det gäller sammansättningen av gruppen bör den enligt Schwaber (2007) bestå av personer som delar samma värdegrund och grundläggande syn på saker och ting. Detta för att det är troligare att det resulterar i mindre konflikter och en mer stimulerande arbetsmiljö, man jobbar med människor som delar ens värderingar, något som också stöds av Harrison (2005). Grupsammansättningen formas givetvis även av Scrumprocessens rekommendationer angående storlek och krav på korsfunktionalitet. Har inte gruppen rätt sammansättning riskerar den att inte kunna nå de uppsatta målen och bidra negativt till gruppens transformation till en självorganiserande enhet (Schwaber 2004).

#### *Organisationens eventuella påtryckningar på Scrum Team*

Som nämnt tidigare, så utövar organisationen i en traditionell arbetsmiljö påtryckningar på arbetsgruppen, i form av orelaterade arbetsuppgifter, ändrade tidsscheman och arbetsuppgifter, samt liknande störande inslag. Ansvaret faller då på ScrumMaster att motverka dessa påtryckningar och därmed tillåta teamet att arbeta vidare enligt deras egen arbetsplan.

Ytterligare en aspekt till detta förhållande är att ScrumMaster måste arbeta för att lära ut och förankra Scrum i organisationen, så att alla involverade är insatta i varför och vad som måste genomföras för att lyckas med Scrum. Förstår individer inte varför, kan individer komma att motarbeta Scrumprocessen och på det viset försena eller förhindra effektivitetsfördelarna som ska uppnås. Det blir helt enkelt en ond cirkel som kan vara svår att komma ur (Schwaber & Beedle 2002). Detta inkluderar även att organisationen i form av avdelningschefer etcetera ska se till att teamet får tillgång till de resurser de behöver för att kunna genomföra projektet så effektivt som möjligt. Teamets sammansättning måste vara så att den kommer klara av projektets uppgifter, teamet ska vara korsfunktionellt.

Organisationen måste även tillhandahålla arbetsmiljö som främjar kommunikation och öppenhet, detta för att få möjlighet att nå Scrums effektivitetsfördelar som dess transparens medför (Schwaber & Beedle 2002). Cockburn (2007) menar att det krävs en öppen arbetsmiljö där teamet sitter nära varandra och att de då får ta del av något han kallar för brus (Noise), vilket innebär att även om man inte är med i en pågående diskussion hör man ändå delar av den och på det viset ökar transparensen inom gruppen. Alla organisationer vill inte låta systemutvecklargrupper arbeta i en miljö som beskrivits ovan och det är då upp till ScrumMaster att försöka få fram en lösning på detta problem, det är som tidigare nämnts han som ska lära ut Scrumprocessen inom organisationen.

#### *Organisationen och ScrumMasters påverkan på grupsammansättningen*

Själva sammansättningen av gruppen är som tidigare nämnts väldigt viktig för att teamet ska komma att lyckas med självorganisation och självbestämmande (Schwaber & Beedle 2002). Gruppen ska vara korsfunktionell, det är en relativt liten systemutvecklargrupp som måste klara av ett brett spektra av uppgifter (Schwaber 2004).

Schwaber (2007) samt Coplien och Harrison (2005) påpekar att ett bra sätt att sätta samman ett team är att välja ut ett par personer som ska vara medlemmar av gruppen och sedan låta dessa välja ut övriga medlemmar. De är troligare att de väljer personer som de vet kommer fungera i grupp, än någon högre upp i organisationen. Cockburn (2007) poängterar dock att detta förhållningsätt till grupsammansättning inte är så lätt att driva igenom, det brukar finnas motstånd inom organisationer mot det. Men han menar även att det ökar chanserna för en lyckad sammansättning. Alltså ökar chansen för en bra sammansatt grupp om organisationen främjar detta sätt att arbeta med grupsammansättning. ScrumMaster påverkar också gruppens sammansättning då denne ofta är delaktig i att välja ut vilka gruppmedlemmar som krävs för att lyckas genomföra ett projekt, det är även han som ska se till att utöka gruppen med lämpliga deltagare om det skulle uppstå problem kopplade till att gruppen saknar någon form av expertis (Schwaber 2007).

#### *Product Owner och dennes koppling till Scrum Team*

Product Owner är nära kopplad till teamet, då han eller hon är ansvarig för utvecklandet av krav för arbetet som ska genomföras, samt skapandet av artefakter så som Product Backlog.

Samarbetet sker i formen av att Product Owner tillsammans med teamet under Sprint Planning Meeting diskuterar vad som ska göras och planerar den kommande sprinten. De krav som Product Owner arbetat fram tas upp till diskussion, där medlemmarna i teamet kan komma med frågor, förslag, och förtydliganden (Schwaber 2004). Förstår inte teamet Product Owners önskemål och bild av det som ska genomföras kommer inte den kommande sprintens resultat bli tillfredställande. Det är då upp till teamet själva att förbättra sitt arbete gentemot Product Owner inför nästa sprint. Schwaber och Beedle (2002) menar att systemutvecklare ofta är ovana att arbeta så nära kunden som Scrum kräver, och det är något som kan påverka resultatet av de första sprintarna. Tills det att teamet själva lärt sig hur de ska arbeta gentemot Product Owner för att nå de uppsatta målen, ligger det i teamets eget intresse att ha ett bra och givande samarbete med Product Owner (Schwaber 2004).

Det finns också problematik kopplat till att Product Owner är så delaktig i Scrumprocessen jämfört med hans delaktighet i exempelvis en sekventiell systemutvecklingsmetod där han har den största delaktigheten i de initiala faserna. Känner teamet att Product Owner inte ger dem det stöd de behöver är det upp till ScrumMaster att se detta hinder och undanröja det, att få Product Owner att inse nyttan av det nära samarbetet och höga delaktigheten (Schwaber 2004). Återigen ligger det på ScrumMasters roll att förankra och lära ut Scrum. Det är också väldigt viktigt att teamet är ärliga och öppna gentemot Product Owner, och inte döljer åsikter och tankar kring det som produceras. Teamet är experter och har varit med om systemutvecklingsprojekt förut och kanske ser problem som inte Product Owner har förutspått (Schwaber & Beedle 2002). Att tillsammans arbeta fram en gemensam vision för arbetet och slutresultatet ser Judy & Krumins-Beens (2008) som en viktig faktor.

#### *ScrumMasters roll i Scrumprocessen, och kopplingen till Scrum Team*

Då Scrum inte fungerar som en vanlig systemutvecklingsmetod med en utnämnd projektledare som bestämmer hur arbetet ska gå till, är förhållandet mellan ScrumMaster och teamet lite annorlunda.

Även om ScrumMaster inte styr detaljer, så kan medlemmar i teamet be honom eller henne om råd, till exempel vid problem i arbetet. ScrumMaster ska även övervaka gruppen och hela processen, och vara vaksam för potentiella problem som gruppen står framför, det kan exempelvis handla om att de är i behov av en extern expert för att kunna komma vidare med arbetet med målen för aktuell sprint (Schwaber 2007). Det är på detta sätt som ScrumMaster guidar teamet, och de i sin tur rådfrågar ScrumMaster (Schwaber 2004). Försöker ScrumMaster detaljstyra teamet kommer han att bidra till att förhindra gruppens transformation till att bli självstyrande och självbestämmande, och på det viset skapa en negativ påverkan på hela projektet (Schwaber 2004). Detta är inte ovanligt om ScrumMaster har lång erfarenhet av traditionell projektstyrning, men det är även något som kan ske naturligt då stressade situationer uppstår menar Schwaber (2007). Han poängterar dock att det är något som både ScrumMaster och teamet måste vara vaksamma på, men ytterst ligger ansvaret på ScrumMaster, han måste låta teamet styra sig själva (Schwaber 2007).

Detta förhållande gäller både kopplingen mellan ScrumMaster och teamet, samt kopplingen mellan ScrumMaster och Scrumprocessen. Detta på grund att ett av ScrumMasters ansvarsområden är att övervaka och reglera Scrumprocessen. Han reglerar den genom att tillsätta resurser, undanröja hinder, vara den som kallar till och håller i möten såsom Daily Scrum etcetera. Han övervakar den genom att vaksamt lyssna på teamet under dessa möten, han övervakar hur arbetet fortskrider genom att undersöka sprintens status via Burndown Charts (Schwaber 2004).

#### *Scrumprocessens arv från det Agila tänkandet*

Scrumprocessen ärver mycket av de tankegångar som återfinns i det agila manifestet. Detta påverkar sedan teamet direkt via faktorer som ökad kommunikation, transparens, etc. Individerna i processen ska dessutom sättas före verktygen. När en systemutvecklargrupp arbetar efter Scrum använder de agila tekniker för att utföra sitt dagliga systemutvecklingsarbete, det kan vara att de använder sig av tekniker såsom Code Review och Pair Programming (Schwaber 2004). Continuous Integration och ett kollektivt ägande av kod är något som krävs för att Scrum ska kunna få så hög transparens som möjligt (Larman 2007).

#### *Teamet och Scrumprocessen*

Teamet påverkas av Scrumprocessen eftersom de måste anpassa sig till arbetssättet som är förknippat med Scrum. De måste följa de procedurer som är till för att skapa transparens i projektet, exempelvis måste de vara noggranna med att fylla i Sprint Backlog, närvara och ärligt berätta om aktuell status på Daily Scrum Meeting. En annan viktig aspekt som handlar om processen och dess transparens är att Scrum Teamet inte får visa upp

inkomplett funktionalitet för Product Owner under Sprint Review Meeting. Då kan Product Owner bli felinformerad om aktuell status, och på det viset undergrävs hans möjlighet att prioritera nästkommande sprint korrekt (Schwaber 2007).

Det handlar även till stor del att teamet måste lära sig självorganisering och självbestämmande, som nämnts tidigare. Det innebär också att teamet accepterar och hänger sig åt att nå de för sprinten uppsatta målen. Detta är teamets huvudfokus och teamet måste inse betydelsen som ligger i att leverera det som de har gått med på att producera. Sprintmål kanske inte kan nås alltid på grund av missförstånd, felaktigt uppskattad komplexitet, naturkrafter etc. Men teamet får inte gå på gång på gång misslyckas med att leverera uppsatta mål, det kan skapa en dålig arbetsmiljö och självkänsla hos teamet (Schwaber & Beedle 2002).

Efter en avklarad sprint har teamet möjlighet att anpassa vilka tekniker de ska använda i nästkommande sprint, tekniker som de anser krävs för att de ska kunna nå uppsatta mål (Schwaber 2004) och inte går stick i stäv med Scrumprocessen (Schwaber & Beedle 2002).

### *Datainsamling*

Innan datainsamlingen kan påbörjas ska ett initialt möte hållas med ScrumMaster och Scrum Teamet. Under detta möte presenteras fallstudien och dess syfte, vilket ger de deltagande personerna möjlighet att ställa frågor angående undersökningen. Det ska även klargöras bakgrunden till projektet de arbetar i, till detta möte ska mötesguiden för bakgrundsinformation användas.. Under detta möte tas anteckningar av lämplig fältarbetare. Man försöker att under mötet läsa av vilka Scrum Team-medlemmar som kan vara lämpliga för att senare bli intervjuade.

Denna fallstudie använder tre olika källor till information; intervjuer med personer som har en betydande roll i det vi undersöker, producerade artefakter samt direktobservation. Det ska genomföras en intervju med ScrumMaster, en med Product Owner samt fyra intervjuer med lämpliga medlemmar i Scrum Teamet. Team-medlemmarna väljs ut efter arbetsbelastning och uppfattning om de är lämpliga för undersökningen eller inte. Intervjuerna genomförs efter respektive intervjuguide och spelas in på bandspelare.

Artefakter som ska samlas in är främst artefakter i form av dokumentation kopplade till Scrum-processen såsom Sprint Backlog. Man undersöker även om det finns andra typer av producerade artefakter under intervjuer och möten. Artefakterna kopieras i den utsträckning organisationen tillåter. Skulle organisationen inte tillåta kopiering tas anteckningar om innehållet av fältarbetaren.

Direktobservationen utförs under ett så kallat Daily Scrum och genomförs efter att intervjuerna har genomförts. Under denna observation tas anteckningar av medverkande fältarbetare.

### *Hantering av insamlad data*

Intervjuerna mot nyckelpersonerna transkriberas med hjälp av en dator till digitalt format. Dessa dokument ges tillbaka till respektive intervjuperson för att denne ska få möjlighet att kontrollera att inga missförstånd skedde under intervjun.

Skulle det finnas anteckningar gällandes insamlade artefakter, kontrollera med organisationen att dessa anteckningar är godkända ur exempelvis sekretesssynpunkt och att ni har rätt att använda anteckningarna. Är anteckningarna handskrivna förs dessa över till digitalt format.

Anteckningar från direktobservation förs över till digitalt format om anteckningarna skulle vara handskrivna.

### *Utformande av rapport*

Rapporten ska utformas efter läsare som inte är vana vid systemutvecklingstermer eller vetenskapliga publikationer och skall vara linjär i sin utformning. På grund av detta krävs förklaring av Scrumprocessen och andemeningen med agil systemutveckling. Tänkt läsare är personer som är intresserade av ämnet och vill veta

mer, systemutvecklare, forskarkollegor, personer i beslutsfattande roll i organisationer som funderar på att implementera Scrum etcetera.

### *Mötesguide, bakgrundsinformation*

Presentera fallstudien och dess syfte, samt hur den kommer att genomföras. Ta upp följande frågor för diskussion.

#### Information angående organisationen

- Information om organisationen
- Använder organisationen Scrum i andra projekt?

#### Information angående projektet

- Bakgrund och beskrivning av projektet
- Hur länge har projektet pågått?
- Vad är det för status på Scrum-processen i dag? Använder de Scrum fullt ut?
- Hur är projektet sammansatt, konsulter, fastanställda?
- Hur lång är en sprint?

#### Frågor

- Har ni några frågor angående fallstudien?

## *Intervjuguide, ScrumMaster*

### Personens bakgrund

- Information om intervjupersonen, namn, tid i organisationen etcetera.
- Hur lång erfarenhet har du av ditt nuvarande yrke?
- Vad har du för tidigare erfarenheter av systemutvecklingsmetoder?
- Vad har du för tidigare erfarenheter av agila tekniker och projekthanteringsmetoder?
- Hur upplever du att dina tidigare erfarenheter påverkar din nuvarande roll?

### Scrumprocessen

- Vad har du fått för utbildning angående Scrum och dess arbetsprocess, vem har givit den?
- På vilket sätt guidar du teamet genom processen?
- Har några problem uppkommit i samband med utbildning inom Scrum till teamet eller andra delar av organisationen?
- Har du upplevt något motstånd mot Scrum och agila tekniker, var fanns detta motstånd i så fall?
- Vad upplever du är fördelarna respektive nackdelarna med ScrumMaster-rollen?
- Hur hanterar ni frågor angående processen och eventuella förändringar av denna idag?
- Kan du ge exempel på råd personer från ditt Scrum Team ber dig om?

### Scrum Team

- Hur sattes teamet ihop?
- Hur ofta förändras sammansättningen av teamet?
- Vad upplever du har varit det svåraste för teamet hittills?
- Upplever du teamet som självorganiserande och självbestämmande?
- Har du sett några skillnader i teamets sätt att arbeta som man kan tillskriva Scrum?
- Kan du ge några exempel på hur teamet löst konflikter?
- Har teamet anammat Scrum, eller har det funnits motstånd? Exempel? (Fyller de i Sprint Backlog etcetera?)
- Hur sker organiseringen inom teamet?
- Kan du ge några exempel på hur teamet fattat beslut tillsammans?
  - Följdfråga: Skillnad mellan strategiska och taktiska beslut?

## *Intervjuguide, Medlem i Scrum Team*

### Personens bakgrund

- Information om intervjupersonen, namn, tid i organisationen etcetera.
- Hur lång erfarenhet har du av ditt nuvarande yrke?
- Vad har du för tidigare erfarenheter av systemutvecklingsmetoder?
- Vad har du för tidigare erfarenheter av agila tekniker och projekthanteringsmetoder och vad är din åsikt angående dessa?
- Hur upplever du att dina tidigare erfarenheter påverkar din nuvarande roll?

### Scrumprocessen

- Vad har du fått för utbildning angående Scrum och dess arbetsprocess, vem har givit den?
- Vilka agila tekniker använder ni i dagsläget?
- Kan du ge några exempel på märkbara skillnader mellan Scrum och andra sätt du arbetat?
- Vilka var enligt dig de största svårigheterna när ni började arbeta efter Scrum?
- Stödjer organisationen ert sätt att arbeta? Kan du ge några exempel på hur de tillhandahåller en bra arbetsmiljö etcetera?
- Hur upplever du att samarbetet med ScrumMaster fungerar? Kontrollerar och beordrar han er eller guidar han er?
- Finns det några svårigheter kopplade till samarbetet med Product Owner?
- Hur tycker du det fungerar att planera en sprint i taget?
- Upplever du att ni efterföljer Scrum-processen, fyller ni till exempel i Sprint Backlog?
- Vad händer om ni inte hinner avklara allt som ska genomföras under aktuell sprint?
- Vad är din åsikt om att arbeta efter Scrum, fördelar, nackdelar?

### Scrum Team

- Kan ni som team påverka vilka som är eller blir medlemmar i teamet?
- Kan du ge några exempel på hur teamet fattat beslut tillsammans?
- Hur sker organiseringen inom teamet?
- Hur kom ni fram till att göra på det sättet från första början?
- Hur upplever du att ert sätt att organisera och utföra ert arbete förändrats sedan starten av projektet?
- Har ni beslutsrätt för att avbryta en sprint?



- Har ni möjlighet att ändra sättet ni kommer att arbeta i nästkommande sprint efter att ni har genomfört ett Sprint Review Meeting?
- Hur fungerar kommunikationen inom ert team? Exempel på problem eller fördelar?
- Upplever du teamet som en relativt homogen grupp? Har ni liknande bakgrund och liknande värderingar
- Har det utvecklats några specifika roller och ansvarsområden i gruppen?
- Har det funnits några konflikter inom gruppen, vilken typ av konflikter och hur har ni löst dem?
- När vänder ni er som team till ScrumMaster?
  - I så fall: Vad har det gällt för problemområden?
- Har du sett några skillnader i teamets sätt att arbeta som man kan tillskriva Scrum?
- Finner du det någonsin frustrerande att arbeta efter Scrum och agila tekniker?
- Vad upplever du som nackdelarna respektive fördelarna med att arbeta i ett Scrum Team?

## *Intervjuguide, Project Owner*

### Personens bakgrund

- Information om intervjupersonen, namn, tid i organisationen, roll etcetera.
- Vad har du för tidigare erfarenheter av att arbeta i ett Scrum-projekt?
- Har du några tidigare erfarenheter av att arbeta deltagande i ett systemutvecklingsprojekt?

### Scrumprocessen

- Vad har du fått för utbildning angående Scrum och dess arbetsprocess, vem har givit den?
- Hur upplever du att det fungerar att arbeta tillsammans med systemutvecklargruppen? Exempel på svårigheter?
- Hur beskriver du din relation gentemot teamet?
- Hur beskriver du din relation gentemot ScrumMaster?
- Har du upplevt att teamet har förändrats allt eftersom iterationerna fortskridit?
- Hur hanterar ni förändring av krav?
- Hur presenteras det arbete som genomförts i en sprint för dig?
- Ser du några nackdelar respektive fördelar med Scrum?

## *Intervjuguide, Linjeförman*

### Projektets bakgrund

- Information om projektet, syfte, hur länge det pågått, hur länge har det körts enligt Scrum, etc.

### Scrumprocessen

- Till vilken grad är du själv inblandad i Scrumprocessen?
- Varför välja Scrum?
- Hur infördes Scrum?
- Hur skedde utbildningen i organisationen kring Scrum?
- Hur skedde sammansättningen av Scrum Team?
- Hur sker förmedlandet av krav i organisationen?
- Vilka agila tekniker använder ni i dagsläget?

### Scrum Team

- Hur påverkas Scrum Team av övriga organisationen?
- Är Scrum Team självorganiserande och självbestämmande?
  - Har de själva skapat interna roller inom gruppen?

## Bilaga 2, Agile Manifesto And Principles

### Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools  
Working software over comprehensive documentation  
Customer collaboration over contract negotiation  
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Kent Beck	James Grenning
Mike Beedle	Jim Highsmith
Arie van Bennekum	Andrew Hunt
Alistair Cockburn	Ron Jeffries
Ward Cunningham	Jon Kern
Martin Fowler	Brian Marick

### Principles behind the Agile Manifesto

*We follow these principles:*

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Business people and developers must work together daily throughout the project.

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Working software is the primary measure of progress.

Agile processes promote sustainable development.  
The sponsors, developers, and users should be able  
to maintain a constant pace indefinitely.

Continuous attention to technical excellence  
and good design enhances agility.

Simplicity--the art of maximizing the amount  
of work not done--is essential.

The best architectures, requirements, and designs  
emerge from self-organizing teams.

At regular intervals, the team reflects on how  
to become more effective, then tunes and adjusts  
its behavior accordingly.

## Bilaga 3, Mall över kodningskategorier

R=Relation  
E=Egenskap

### Relation

R Relation

R: Organisationen utövar påtryckningar

R: Organisationen påverkar sammansättningen

R: ScrumMaster skyddar ScrumTeam

R: ScrumMaster påverkar sammansättningen

R: ScrumMaster lär ut och förankrar Scrum

R: ScrumMaster guidar Scrum Team

R: ScrumMaster guidar och lär ut till ScrumTeam

R: ScrumMaster övervakar och reglerar Scrumprocessen

R: Samarbetar mellan Product Owner och ScrumTeam

R: Product Owner förmedlar krav till ScrumTeam

R: ScrumTeam rådfrågar ScrumMaster

R: ScrumTeam påverkar och påverkas av Scrumprocessen

R: Scrumprocessen skapar gruppfaktorer

R: Agile & XP formar iterationer

R: ScrumMaster lär ut Scrum till Product Owner

R-Org-Påtryck

R-Org-PåverkSamman

R-SM-SkyddarTeam

R-SM-PåverkarSamman

R-SM-LärUtScrum

R-SM-GuidarTeam

R-SM-GuidarLärUtTeam

R-SM-ÖverRegProcess

R-PO-ST-Samarbete

R-PO-FörmKravTeam

R-ST-RådfrågarSM

R-ST-SP-Påverkan

R-SP-SkaparGruppFakt

R-Agile-FormarIter

R-SM-LärUtScrumPO

### Egenskap E

E: Organisation främjar arbetsmiljö

E: Organisation tillhandahåller resurser

E: ScrumMaster undviker detaljstyrning

E: ScrumMaster tidigare erfarenheter

E: ScrumMaster expert på Scrumprocessen

E: ScrumProcess har transparens

E: Individuella faktorer, teknik och processkunskap

E: Individuella faktorer, tidigare erfarenheter

E: Individuella faktorer, hög yrkesskicklighet

E: Gruppfaktorer, självorganiserande och självbestämmande

E: Gruppfaktorer, kommunikation och öppenhet

E: Gruppfaktorer, samarbete

E: Gruppfaktorer, grupp-sammansättning

E: Gruppfaktorer, konflikter

E: Scrumprocess övervakas via verktyg

E: ScrumMaster måste hålla isär och separera roller

E: Product Owner måste kunna släppa kontroll

E: ScrumTeam tar kollektivt ansvar

E-Org-Främjar

E-Org-Resurs

E-SM-Detaljstyr

E-SM-TidigareErf

E-SM-ExpertScrum

E-Scrum-Transp

E-Ind-TekProc

E-Ind-TidigareErf

E-Ind-HögYrkes

E-Grupp-SjälvOrgBest

E-Grupp-Komm

E-Grupp-Samarb

E-Grupp-Sammans

E-Grupp-Konflikt

E-SP-Verktyg

E-SM-SepRoller

E-PO-SläppaKontr

E-ST-KollektivtAnsv

## Bilaga 4, Exempel på kodning

<E-SP-Verktyg>	MS	Du nämnde tidigare att agila principer kräver mer av utvecklarna, att det krävde effektivare och bättre utvecklare. Har du upplevt det problemet här? Att vissa kanske är experter?	Då får man ju det problemet, att de kanske inte tror på timmarna för det tror det ska ta längre tid när de börjar. Då försökte vi hjälpa dem och så, men det är jättesvårt. Det vore bra att få fram en formel för att när en ny person kommer in hur han tidsestimerar och hur man löser det. Dels tidsestimerar åt honom är svårt för man vet inte hur lång tid han tar. Dels tar det resurser från de andra, en viss procent då, och det vet vi ju inte, det kommer bli jättesvårt den här sprinten, det kommer bli jättesvårt, vi får se, som ett försök nu denna sprinten.
<E-Scrum-Transp>	LA	Har du sett några skillnader i teamets sätt att arbeta som du tror att man kan tillskriva Scrum?	Ja, man ser ett slut på saker och ting, som när man har en deadline, så har man ambitionen att bli färdiga till dess, om man bara har ett löpande projekt så är det mer "jaja, om detta tar två veckor eller en gör ju inte så mycket". Men när vi nu efter två veckor ska ha någonting att visa.. Just det här att ha något att visa har en stor effekt, man vill ju visa något som är bra
<E-PO-SläppaKontr>	NH	Hur tror du det hade blivit om du suttit i ett annat hus här?	Då tror jag att jag hade haft mindre insikt såklart i den dagliga verksamheten och haft större krav på vad som levereras. Jag tror också att jag skulle kräva in mer statusrapporter från ScrumMaster, för nu har vi ju mycket omedveten kommunikation, och varje gång man hör någon säga något så stannar man ju upp och bekräftar det och sen så är det klart liksom

## **Bilaga 5, Direktobservation: Daily Scrum nr 1**

Alla deltagarna står upp, i ett rum fyllt med utrustning, och inga stolar.

ScrumMaster och teammedlemmarna är närvarande, men inte Product Owner.

ScrumMaster inleder mötet genom att gå rakt på sak och frågar den första medlemmen i teamet om en statusrapport.

I tur och ordning går alla medlemmarna igenom vad de gjort sen sist, vad som ska göras, och vad som är problematiskt. I detta steg sker viss tidsestimering av framtida arbetsuppgifter. ScrumMaster tipsar vid ett par items om att även uppdatera tidsestimeringen i systemet som används för administration av Scrum.

Hela teamet är involverat i att arbeta fram en plan för hur problem ska lösas, och detta sker i form av antingen korta tips på hur något kan lösas, eller genom att förklara vem utanför mötet som kan tänkas ha lösningen.

Det är teammedlemmarna som argumenterar sinsemellan om de olika items som tas upp, och ScrumMaster övervakar och ger input där det behövs.

Viss diskussion sker kring fördelning av arbete, och hur det ska ske på bästa sätt.

Nära slutet av mötet visar ScrumMaster upp ett Burndown Chart som nästan exakt matchar den uttänkta linjen för projektet, och deltagarna i mötet ger synpunkter på den enda avvikelser som finns där, samt hur arbetet ska genomföras för att fortsätta att prestera på samma nivå.

Sist av allt dyker lite allmänna kommentarer upp, och mötet avslutas, allting på under 10 minuter.



## **Bilaga 6, Direktobservation Daily Scrum nr 2**

Alla deltagarna står upp, i ett rum fyllt med utrustning, och inga stolar.

ScrumMaster och teammedlemmarna är närvarande, men inte Product Owner.

ScrumMaster inleder mötet genom att gå rakt på sak och frågar den första medlemmen i teamet om en statusrapport.

I tur och ordning går alla medlemmarna igenom vad de gjort sen sist, vad som ska göras, och vad som är problematiskt. Det varierar mycket i tid mellan de olika medlemmarnas genomgångar.

ScrumMaster är mycket aktiv i att ge tips, och detta leder till diskussion mellan medlemmarna om lösningar, mest i form av vem som ska kontaktas efter mötet. Mycket av detta handlar om vem som vet vad i teamets omgivning.

ScrumMaster har koll på de olika "impedements" som existerar i den aktuella Sprinten, och tar upp dessa kort under mötet. Med ett impedement menades ett hinder som stod i vägen för att ScrumTeamet skulle kunna genomföra vissa uppgifter.

Vid ett större problem dyker det upp att problemet stötts på tidigare, och två medlemmar ur teamet kommer överens om att ha ett extramöte efter Daily Scrum för att lösa problemet, detta stöds av ScrumMaster.

ScrumMaster visar sedan upp gruppens aktuella Burndown Chart tillsammans med ett annat Burndown Chart för jämförelse, och gruppen kommenterar över vissa avvikelser och hur de lösts.

Allting går på under 10 minuter.

## **Bilaga 7, Observationer i arbetsmiljön**

Under våra besök på SonyEricsson kunde vi göra följande observationer:

Arbetsmiljön var väldigt öppen, och medlemmarna i teamet satt nära varandra. ScrumMaster och Product Owner hade arbetsstationer som vilken medlem som helst. Även linjeföraren hade en sådan placering.

ScrumMaster och Product Owner satt bredvid varandra, och kunde således lätt utbyta frågor och kommentarer, vilket kunde observeras att de gjorde.

Avdelningen var avskärmd från närliggande avdelningar, och det verkade inte finnas några yttre störningar.

Vid flera tillfällen observerades att teammedlemmar samlades mer än två personer för att diskutera ihop lösningar och liknande. Att teammedlemmar diskuterade två och två var ännu vanligare. Vi observerade även att två teammedlemmar hade en diskussion, och helt plötsligt nämner teammedlem Jonas Nyberg en sak för dem som diskuterar, och han blir en aktiv deltagare i diskussionen.

Flera olika samtal om arbetet kunde urskiljas, och om man så hade velat, hade det gått att följa dem. Vissa i arbetsmiljön gjorde detta genom att lyfta huvudena från sina arbeten. Vissa teammedlemmar sågs ibland sittandes med stora hörlurar, och de upplevdes som väldigt koncentrerade.

Vid minst ett tillfälle kunde observeras att en teammedlem gick till ScrumMaster för att fråga om råd.

Det fanns två skärmar uppsatta, tydligt synliga för alla på avdelningen. Den ena skärmen visade den aktuella sprintens Burndown Charts, den andra visade statusen över Continuous Integration på avdelningens olika projekt.

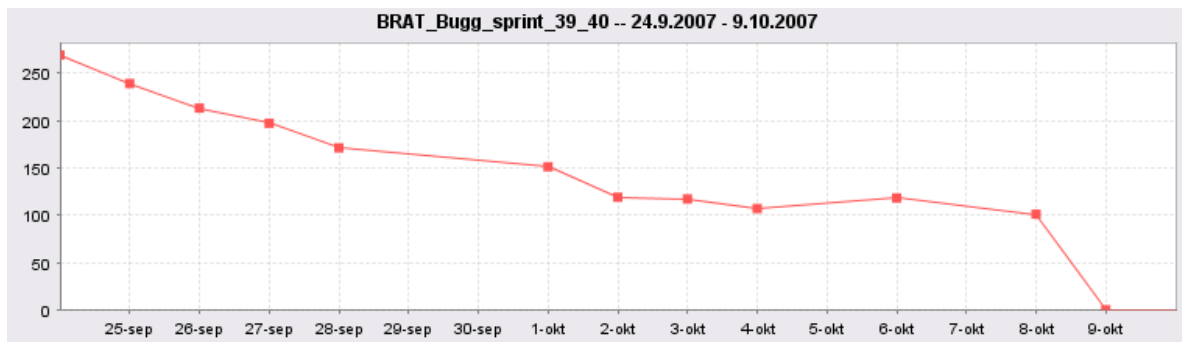
# Bilaga 8, Utdrag av Sprint Backlog

24/9 2007 – 9/10 2007

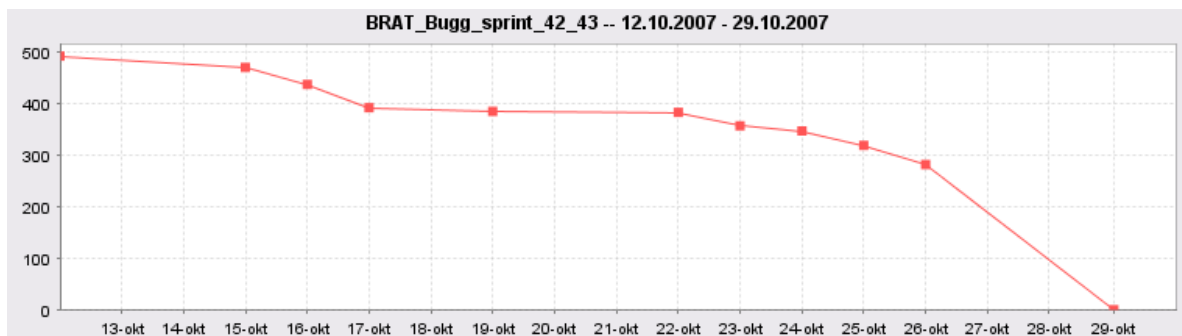
A	B	C	D	E	F	G	H	I				
Pri.	Backlog Item	Task Title	Point Person	Task Statu	24	9	25	9	26	9	27	9
1	DMS00148864: Error when reading screen: Not OK responds to AT command AT*ZIP=240,0,179,0	Response Update Use Case	Fredrik Darinder	Done	1	1	0					
2	DMS00148864: Error when reading screen: Not OK responds to AT command AT*ZIP=240,0,179,0	Response Write Test Case	Fredrik Darinder	Done	1	1	0					
3	DMS00148864: Error when reading screen: Not OK responds to AT command AT*ZIP=240,0,179,0	Response Implement	Fredrik Darinder	Done	4	5	1					
4	DMS00148864: Error when reading screen: Not OK responds to AT command AT*ZIP=240,0,179,0	Response Test	Fredrik Darinder	Done	2		1					
5	DMS00207809: Unhandled exception when closing application	Update Use Case	Fredrik Darinder	Done	1	0						
6	DMS00207809: Unhandled exception when closing application	Write Test Case	Fredrik Darinder	Done	2	0						
7	DMS00207809: Unhandled exception when closing application	Test	Fredrik Darinder	Done	2	0						
8	DMS00207809: Unhandled exception when closing application	Implement	Fredrik Darinder	Done	1	0						
9	DMS00207194: BRAT: MenuNavigation/Wait fails every time	Investigate	Magnus Rosling	Done	4	4	4					
10	DMS00207194: BRAT: MenuNavigation/Wait fails every time	Test	Anders Nyberg	Done	1							
11	DMS00189178: Test steps are being doubled when addin a test step provider plugin	Update Use Case	Jonas Nyberg	Done	1	1	1					
12	DMS00189178: Test steps are being doubled when addin a test step provider plugin	Write Test Case	Jonas Nyberg	Done	1	1	1					
13	DMS00189178: Test steps are being doubled when addin a test step provider plugin	Test	Jonas Nyberg	Done	2		2					
14	DMS00198782: The Add button does not work.	Update Use Case	Jonas Nyberg	Done	1	0						
15	DMS00198782: The Add button does not work.	Write Test Case	Jonas Nyberg	Done	1	0	0					
16	DMS00198782: The Add button does not work.	Test	Fredrik Darinder	Done	2	0	1					
17	DMS00200960: BRAT Crashes when the same phone is used twice when closing the com ports	Update Use Case	Jonas Nyberg	Done	1							
18	DMS00200960: BRAT Crashes when the same phone is used twice when closing the com ports	Write Test Case	Jonas Nyberg	Done	1	1						
19	DMS00200960: BRAT Crashes when the same phone is used twice when closing the com ports	Test	Jonas Nyberg	Done	2							
20	DMS00207160: Unlinked parameters are coloured with the same colour as linked	Update Use Case	Lars Akerman	Done	1	0						
21	DMS00207160: Unlinked parameters are coloured with the same colour as linked	Write Test Case	Lars Akerman	Done	1	0						1
22	DMS00207160: Unlinked parameters are coloured with the same colour as linked	Implement	Lars Akerman	Done	4	3	0					
23	DMS00207160: Unlinked parameters are coloured with the same colour as linked	Test	Jonas Nyberg	Done	2	2	2					
24	DMS00210138: Saving Advanced Settings for not connected equipment throws NullPointerException	Update Use Case	Fredrik Darinder	Done	1	1	0					
25	DMS00210138: Saving Advanced Settings for not connected equipment throws NullPointerException	Write Test Case	Fredrik Darinder	Done	1	1	0					
26	DMS00210138: Saving Advanced Settings for not connected equipment throws NullPointerException	Test	Fredrik Darinder	Done	2	0						
27	DMS00211069: ExecutionEngine doesn't write to the shared log	Update Use Case	Magnus Rosling	Done	1	1	0					
28	DMS00211069: ExecutionEngine doesn't write to the shared log	Write Test Case	Magnus Rosling	Done	1	1	0					
29	DMS00211069: ExecutionEngine doesn't write to the shared log	Implement	Magnus Rosling	Done	2	2	0					
30	DMS00211069: ExecutionEngine doesn't write to the shared log	Test	Stellan Vibeig	Done	2		2					0

XX

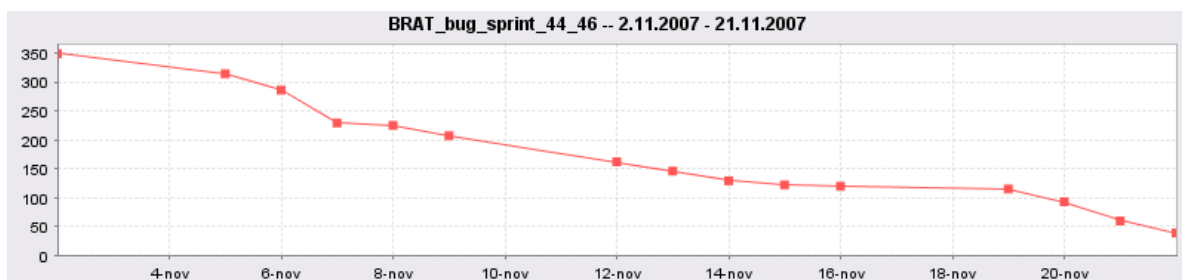
## Bilaga 9, Burndown Charts



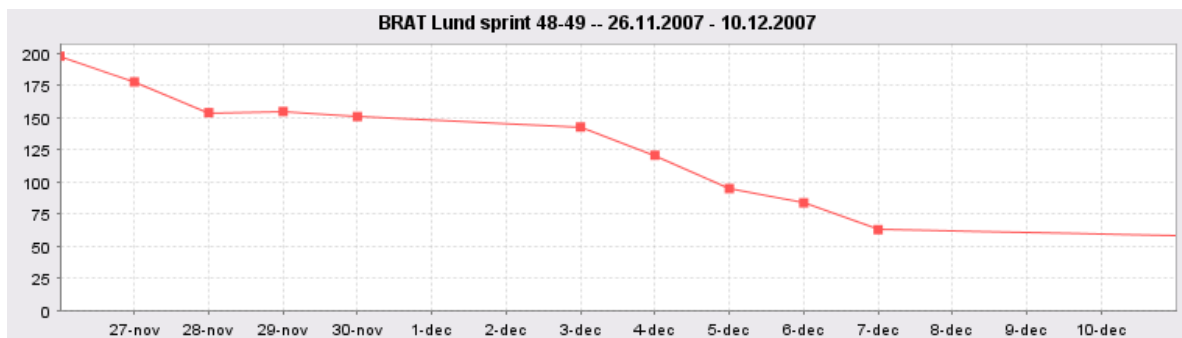
Vecka 39-40, 2007



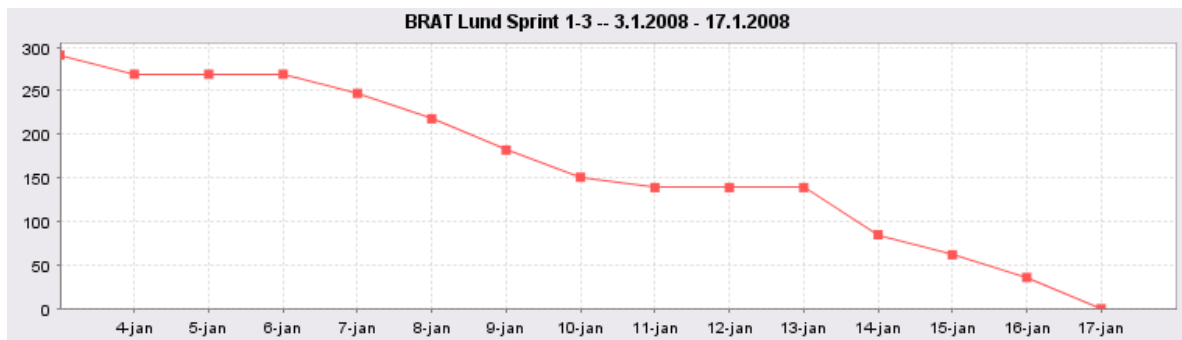
Vecka 42-43, 2007



Vecka 44-46, 2007



Vecka 48-49, 2007



Vecka 1-3, 2008

I Burndown Charts för vecka 39-40 samt 42-43 plockades uppgifter ur Sprint Backlog bort innan sprinten avslutades, vilket resulterar i att de ser ut som de hann klart med sprinten. Detta berodde enligt linjeföraren på att man inte visste hur man skulle hantera Sprint Backlog ifall man inte hann klart med alla uppgifter i en sprint. Resultatet blir att dessa två Burndown Charts ser bättre ut än de egentligen var.

## Bilaga 10, Anders Nyberg, Linjechef

### Intervjuare:

**Intervju på SonyEricsson den 22:a januari klockan 15.30 med Anders Nyberg, linjechef på det Scrum-projekt vi ska undersöka. Tack för att du tar dig tid. Vi tänkte börja med lite grundläggande frågor och du kan väl ta en bakgrundsbeskrivning av projektet.**

AN:

Ja det är som sagt ett testautomationsverktyg som används då internt på SonyEricsson, så det är bara interna kunder, som bygger på att man ska simulera en användare av en mobiltelefon. Så det har ett protokoll vilket man skickar kommandon via en vanlig serieport egentligen, en USBsladd till telefonen som man då kan simulera knapptryckningar och läsa vad som står på displayen osv. Vi använder ett skriptspråk som heter Ticet, applikationen är skriven i .NET, 2.0 och C#. Det som är lite speciellt med systemet är det att vi inte använder ett rent skriptspråk, tryck vänsterknappen ner tre gånger osv. utan det är lite mer modellbaserat. Vi har en XML-modell nu gentemot en telefon som man navigerar mot, som beskriver hur man hittar till olika ställen i telefonen, hur strukturen ser ut, vilka features den har. Så scripten är på en lite högre logisk abstraktionsnivå.

### Intervjuare:

**Hur länge har applikationen utvecklats?**

AN:

Det började då som ett hobbyprojekt som jag och en till kille gjorde när vi jobbade som testare. Vad kan det varit sen, två år sedan kanske. Vi satt en två tre stycken med det på den testavdelningen och började använda det mer, det växte och blev större. Sen blev det ett officiellt projekt och vi var kanske en fem sex personer. Sen efter en omorganisation för ett år sedan ungefär, så blev det här en särskild testverktygsavdelning som jag är linjechef för. Som då servar hela Global Product \*ohörbart\*, GPW, som är vår stora produktutvecklingsavdelning på SonyEricsson. Och nu driver vi BRAT under den linjeorganisationen och har en del andra småprojekt och så.

### Intervjuare:

**Varför valde du att ni skulle köra Scrum eller var det någon annan?**

AN:

Ja det var jag och SonyEricsson är som sagt ett väldigt dynamiskt företag och förutom när vi satt med systemtest så körde vi väldigt Agile, eller i den stilen, vi visste inte ens om att vi körde Agile. När vi satt och utvecklade så satt vi nära och fick feedback, och det var väldigt effektivt att köra så. När vi flyttades ut märkte vi att vi förlorade kontakten med folk och då blev det svårt att få reda på vad folk egentligen ville ha. Vi har massa features som vi underhåller som ingen använder kanske, massa skräp där osv. Då var jag på ett föredrag, jag tror det var datorföreningen i Skåne som höll, just om Scrum då, och de berättar om Scrum och det verkade väldigt käckt och verkade vara en bra sak. Då tänkte jag att vi skulle prova det, här då. Och så förde vi in det.

### Intervjuare:

**Hur började ni sen då, började ni bara rakt av, kollade ni litteratur, hade ni någon utbildning då vid start?**

AN:

Utbildningen bestod i att kolla litteratur. Och jag och en kille då, som inte jobbar här längre som heter Johan förde vidare till de andra, efter att vi läst på och pratat om hur det skulle funka. Och förde in det då.

### Intervjuare:

**Och det här var tre månader sen då eller?**

AN:

Nej det var nog lite längre sen, fem kanske som vi började då. Vi använde först ett Excel-ark som vi använde för att hålla koll på Scrum.

**Intervjuare:**

**Använder ni några verktyg för Scrum då alltså?**

AN:

Ja idag använder vi ScrumWorks, deras gratisvariant som finns. Tillsammans med de vanliga systemet på SonyEricsson, det vill säga DMS som bygger på ClearQuest som är ett issue-hanteringssystem. Vi använder ClearCase vid versionshantering. Så använder vi HP, som hette Mercury innan för att beskriva testfall, use-cases och sådant.

**Intervjuare:**

**Vad är det för status på Scrumprojekten idag, ni nämnde tidigare på introduktionen att ni hade två Scrumgrupper.**

AN:

Mmm. Status är då att vi började med Scrum här, och det var inte 100% Scrum då, eller ja det var det i och för sig men vi var inte utbildade på det så vi kanske gjorde fel någon gång. Vi körde sprintar från start, teamet var med och tidsuppskattade, vi försökte hålla teamen skyddade under sprintarna. Det var det jag gillade med Scrum, att man håller teamet skyddat, och så vidare. Men i början var det att vi inte hanterade 100% av sprinten, vi nådde nästan alltid målet men inte riktigt. Då var det ju ett Scrumteam som körde, med 7 personer, och så fanns det 3 övriga personer som drev lite andra småprojekt. De körde i och för sig lite Scrum-aktigt på ett annat projekt. Sen i början av januari, för inte alls länge sedan var vår ScrumMaster, Lars, på en utbildning, en riktig ScrumMaster-utbildning, hos SoftHouse, innan dess kanske man kan se det som att jag även hade en ScrumMasterroll, men efter det kom han ju tillbaka med en massa tips, vi var ju rätt på det men det fanns en del småsaker som man skulle kunna fixa till. Och då beslutade vi att snart sätta ihop två Scrumteam, för att försöka hålla dem intakta.

**Intervjuare:**

**Vad var det för problem som han såg efter utbildningen?**

AN:

Ett problem var ju att teamet inte uppskattade Product Backlogen, alltså de estimerade på Sprint Planning Meeting. Men att uppskatta komplexitetspoängen i Product Backloge gjorde de inte utan vi hade tolkat det som att Product Backlogen är i dagar som estimerat och den satt Product Owner och höll på med då och satt och pillade i rätt mycket, så det var ju en stor skillnad. Sen mycket mycket hårdare på att en sprint ska bli klar, det är inget snack om saken, det är målet som teamet har. Och att vi skulle börja lägga in sånt som helger i sprinten för det gjorde vi inte tidigare utan det är klassificerat på en burnchart-nivå, man ska lägga in utbildningar, helger, semester. Innan så tog vi ju bort saker om en av medlemmarna skulle vara borta, det tog vi med i beräkningarna under sprinten så att de skulle hinna med det vi planerade för sprinten i och med det.

**Intervjuare:**

**Kan man säga det så då att själva utvecklingarna i Scrumteamet bestämmer lite mer nu,än vad de gjorde tidigare eftersom de är mer med och estimerar?**

AN:

Ja de var ju med och estimerade sprintarna men inte komplexiteten i Product Backlog, så visst de är ju med mer och bestämmer. Så det är väl de två stora skillnaderna som Lars tyckte att vi måste skärpa till oss, och det var ju den här komplexitetspoängen som vi då hade feltolkat.

**Intervjuare:**

**Har ni haft problem med att hålla teamet intakt under sprintarna? Har du sådan makt över resurserna att du kan säga, att nu ska de här fyra sitta med det här och det under den här sprinten.**

AN:

Nej det har inte varit något problem faktiskt. Det tycker jag nog har fungerat bra. Det är nog för vi har planerat utifrån att en person, en ScrumTeam-medlem har sex produktiva timmar per dag och inte åtta. Vi har tagit med att folk ibland måste springa på ett möte, måste kolla sina mail, vad det nu kan vara. Så vi räknar med det, 6 timmars produktiv tid per dag.

**Intervjuare:**

**Hur är det med teamet, har det varit intakt sedan början? Är det någon som har lämnat?**

AN:

Jodå vi hade till exempel en kille innan jul, som var en av våra tyngsta utvecklare som hade varit med från början som lämnade teamet. Men inte jättemycket ändringar, men vissa förändringar har det varit.

**Intervjuare:**

**Har du märkt några problem med att någon har lämnat gruppen, som den är killen?**

AN:

Nej förvånansvärt lite.

**Intervjuare:**

**Ok, hur länge kör ni en sprint?**

AN:

2 veckor.

**Intervjuare:**

**Du sa tidigare att teamet består av både konsulter och fastanställda, har det påverkat någonting eller har ni kanske någon gång känt att ni behöver ha in en extra resurs och då har ni då tagit in en konsult till?**

AN:

Nej konsultsituationen inom SonyEricsson är väldigt speciell nu, det är ju väldigt mycket resursförstärkning, så de är ju som vanliga anställda i många aspekter, de sitter hos oss i långa perioder och är inte inhyrda för att göra en speciell uppgift och sedan ska de ut, utan man hyr in dem på ett år. Så det har inte varit några problem då.

**Intervjuare:**

**Tacksam situation på företaget då kanske?**

AN:

Ja absolut!

**Intervjuare:**

**Ni ska ju nu ha två Scrumteam, delar de då samma arbetsmiljö och jobbar de med samma produkter eller?**

AN:

Ja eller olika produkter, just nu har vi BRAT som ett stort projekt och det är ju då ganska intensivt, vi har precis hållit på med en stor Refactoring under lång tid, för det var ju då ett hobbyprojekt som växte och kanske inte var så strukturerat innan och skrivit om det från 1.1, vi gjorde en switch där, så just nu kommer båda ScrumTeamen jobba på BRAT. Tanken jag har är att inom långa loppet så ska ett team vara BRAT number one så att säga, och framförallt jobba med det. Och det andra ska jobba med BRAT när det behövs eller så skapar vi en annan produkt som vi kallar för Others, slasktratt. Det kan vara mycket småsaker, en buggfix som tar tre dagar som är det som behöver göras. Och det är ju då det andra teamets uppgift och den Product Backloggen kommer jag då att sköta.

**Intevjuare:**

**Men båda teamen sitter i samma miljö?**

AN:

Japp.

**Intevjuare:**

**Och det fungerar bra, inga problem med delat miljö för olika projekt?**

AN:

Nej.



**Intervjuare:**

**Är det en Windows- eller webb-applikation?**

AN:

Det är en Windows-applikation men det skulle kunna vara en webbapplikation som vi gör i framtiden för andra saker. Det är ju inte bara BRAT egentligen utan det är ju kanske andra applikationer organisationen behöver.

**Intervjuare:**

**Vem kommer med krav idag, eller ehh vem skapar själva kravbilden idag, är det era testare som kommer med förslag?**

AN:

Oftast är det testare.

**Intervjuare:**

**Hur får ni in dem?**

AN:

Det är vi DMS oftast, ibland får man ett e-mail som säger, vi vill ha det här, men då vill vi att de lägger en DMS på det. Så vi får en spårbarhet på det. Då lägger det en sådan som beskriver något de vill ha och då uppdaterar vi vårt Use Case som vi har liggande, det är den kravbild vi har. Och sedan genomför det då i en sprint, skapar Use Caset. Det är en sorts Testdriven development fast inte med hjälp av Unit Test utan vi börjar med att skriva Use Caset och testfallen och sen så implementeras det och sen testas testarna det under sprinten och så släpper vi en ny version.

**Intervjuare:**

**Ok, så ni använder en form av TDD där ni testas i efterhand?**

AN:

Nej du skriver testfallen först, det är som jag sa, det är ett stort gammal system så vi har inte börjat på bred front med Unit Test, utan det är kanske TDD fast manuellt. Men du skriver ju Use Case och testfall innan du börjar kodar, så du börjar inte med att koda.

**Intervjuare:**

**Går testarnas nya krav direkt till teamet eller via product owner?**

AN:

Det går via product owner.

**Intervjuare:**

**Hur har ni utbildat de som ingår i teamet? Har ni haft en speciell workshop där ni har pratat om vad ni vill använda och hur, har de kunnat påverka det?**

AN:

Ja som vi sa innan så hade vi mycket snack innan vi körde igång, och så har vi Sprint Retrospective efter varje sprint. Där vi har tagit upp mycket, ofta små enkla saker har vi påverkat. Till exempel en sak som när de tidsuppskattar vill de ha, nu när det är mycket buggfixning så får vi egentligen alla buggar och de ser ungefär likadana ut, vad som behöver göras då, och det är ju inte alltid saker som behövs som är med. Tex olika tasks, som Reproduce på en bugg, och det märker man att det tog längre tid. Och då märkte vi att ja, vi vill jobba och ha en Reproduce task också, och den typen av feedback har vi tagit upp.

**Intervjuare:**

**Använder ni några fler agila tekniker?**

AN:

Vi har börjat med Continuous Integration och då Unit Testning, som inte BRAT har kört så länge ännu men de andra småprojekten har kört det desto mera. Han Dick som ni kommer intervjuar har jobbat med en produkt som heter On The Road, där har han kört väldigt mycket Unit Test.

**Intervjuare:**

**Om ni har Continuous Integration har ni kollektivt ägande av kod fullt ut?**

AN:

Teamet kör på samma branch. Förutom det här Scrumish-teamet i Kina, de har inte det av infrastrukterskäl.

**Intervjuare:**

**Har ni haft detta innan eller var det något ni började när ni började med Scrum?**

AN:

Nej från första början körde vi med att alla utvecklare hade en branch. Men då var vi inte lika många som vi är nu. Men om vi pratar om tre fyra personer hade vi en mer klassisk approach där alla hade varsin branch.

**Intervjuare:**

**Om man kollar på Continuous Integration, har ni haft några problem kopplade till det. Det finns kanske vissa människor som har svårt för att dela med sig av kod, lite ”du ska inte ändra i min kod”-mentalitet?**

AN:

Nej något sådant har vi inte haft alls, det kan jag inte påstå. Vi har kört ganska hårt på det även innan vi har kört Scrum. Vi har Coding Guidelines, och det är inte förhandlingsbart. Och vi har Code Reviews som vi har kört hela tiden. Vi är väldigt vana vid att folk tittar på din kod och tycker till och ger kommentarer, så det har inte varit något sådant tjafs.

**Intervjuare:**

**Hur ofta och vem gör en Code Review?**

AN:

Om vi tar Scrum då så är det något som de som kör det har förändrat. Innan körde vi så att vi hade en stor Code Review chunk i slutet, men så gör vi inte längre. Utan för varje implementation de ska göra har de en Code Review Task kopplat till den implementationen. Som de inte själva får göra, så det får inte vara någon som skrivit koden som granskar den, så de gör Code Review hela tiden gentemot varandra.

**Intervjuare:**

**Har du någon uppfattning om det finns några uttalade roller inom själva ScrumTeamet? Kanske någon som är expert på databaser som alltid gör de sakerna?**

AN:

Ja på ett sätt har vi det, framförallt har vi David då som är vår CM-människa, det är han som svarar på sådana frågor och sköter det. Vi har en annan kille, Karl som ännu inte är en fullfjädrad utvecklare för han kommer från test från början, han har jobbat väldigt mycket med att skriva Use Case och testfall och visst han kodar en del men inte de mest tunga grejerna. Om man kollar på Magnus och Dick som kör sitt egna lilla projekt, där var det Magnus som hade det som sitt skötebarn från början så han är domänexpert. Då har Dick jobbar väldigt mycket med Unit Test osv.

**Intervjuare:**

**Är det roller teamet själva har arbetat fram?**

AN:

Min approach är ungefär, att när de kommer till mig och säger att de skulle vilja ha någon som är ansvarig för att uppdatera testfallen. ”Ja, gör det då” menar jag då.

**Intervjuare:**

**Vad kommer de och frågar dig om exempel?**

AN:

Nu är det fortfarande ganska mycket domänfrågor eftersom jag är domänexpert, det är ju min bebis från början så tyvärr är jag nog domänexpert fortfarande. Men det är något jag försöker komma ifrån.

**Intervjuare:**

**Fungerar du lite som en Product Owner då kan man säga?**

AN:

Ja det gör jag ju i viss mån också, Niklas kan inte produkten lika bra som jag kan så jag är med där ibland och hjälper också. Men jag är inte med på Sprint Planning det är jag inte utan det har jag gått ifrån. Utan det är mer praktiska saker, som att "jag vill vara ledig i morgon, kan jag vara det?". Och då är mitt svar, att så länge det inte påverkar teamet och sprinten så är det ok.

**Intervjuare:**

**Om utvecklarna hade kommit och sagt, det här fungerar inte, vi vill inte köra Scrum, hur hade du reagerat då tror du?**

AN:

Då hade jag frågat varför det inte fungerar, berätta för mig vad ni vill förändra? Alltså alla förslag vi får från utvecklarna, idéer om arbetssätt och så säger jag att vi provar. Sen vid Sprint Retrospective kollar vi på det, var det bra? Nej? I så fall skiter vi i det.

**Intervjuare:**

**Har du någon över dig som kommer med åsikter om hur ni arbetar?**

AN:

Inte om hur vi arbetar, utan snarare om hur vi presterar. Annars har jag inte sådant.

**Intervjuare:**

**Har du märkt någon produktivitetsförändring sen ni gick över kanske till mer "korrekt" Scrum, inte så mycket cowboy-stil?**

AN:

Svårt att säga eftersom det skedde samtidigt som en omorganisation, med många nya utvecklare. Vad jag känner är att det är en jätteskillnad i struktur, de vet vad de ska göra och det har koll på hur man gör. Innan så hade jag koll på allting och de var yra höns. Hjälpporganisationen är borta. Det är den stora skillnaden, teamet tar mycket större eget ansvar och sköter sig själva.

**Intervjuare:**

**Skulle du säga att teamet är självbestämmande och självorganiserande fullt ut eller är det någonting som håller på att transformeras?**

AN:

Domänexpertproblemet gör att det fortfarande är en övergångsfas men de är i stort sett självorganiserande. Det är ju så jag ställer krav på dem också, är de inte färdiga i tid så är inte min reaktion varför har ni tagit så lång tid, min reaktion är varför har ni tidsuppskattat så dåligt? Jag håller dem ansvariga som team, om de säger att de är sena för att Karl är på semester så undrar jag varför de inte tog med det i planeringen, för det köper jag inte. Utan nästa gång ska ni ta med det i beräkningen istället. Vi hade en gång då en i teamet bytte dator mitt i en sprint och det var något som vi märkte inte var ok. Så det är att jag ser på teamet som en person. Det var därför jag gillade Scrum, jag har sett lite olika saker. Allt från eget ansvar, till jättestora grupper och så vidare, men det var då toppstyrt. Och jag tänkte då med Scrum, om jag är på krogen med mina fem bästa polare har inte vi en utsedd projektledare som bestämmer var vi ska gå någonstans. Det är något som vi löser själva, men när vi är femton blir det något som är jobbigt. Så där har Scrum hittat den magiska gränsen, det är mellan fem och åtta personer en grupp klarar av för att faktiskt vara självorganiserande. Och det är det som också är fint med Scrum, att under en sprint då säger man att under två veckor har ni den här sandlådan att jobba inom. Så länge ni gör det så får ni göra precis hur ni vill. Men går ni utanför den, då är det inte ok längre då måste ni kolla med mig. Det är det som jag gillar med Scrum.

**Intervjuare:**

**Varför valde ni två veckor som sprintlängd?**

AN:

Därför det här är ett sådant förbaskat snabbt företag. Väldigt mycket Refactoring, släppa nya releaser, mycket

brandkårsutryckning. ”Jäklar blev det så där, hoppsan, så fyra veckor hade vart, jag vet inte, jag vill komma bort från det här”. Vi hade ett läge där vi gjorde ungefär bara brandkårsutryckningar och då får man ingenting gjort. Samtidigt tror jag inte användarna accepterar att ja börja förbereda för denna featuren i nästa sprint, så blir den implementerad och levererad om två månader. Men två veckor, det finns inte många saker som är så bråttom att inte det fungerar. Men det kan finnas lägen då man kanske går ifrån detta men då blir man kanske mindre produktiva.

**Intervjuare:**

**Under en sprint, vad har ni för regler för att checka in kod, vet du vad ni har som definerar en done?**

AN:

Det ska vara testat.

**Intervjuare:**

**I vilken utsträckning är det något som du har varit med och fört in?**

AN:

Vi pratade tidigare om att om det skulle vara en bugg, så är det ett par fasta tasks. Du har reproduce, update use case, write test case, implement, code review och test. Har en bugg gått igenom detta, då är den färdig. Ser de en DMS ska de inte reagera över hur de ska fixa den utan även kolla hur kommer det sig att vi har släppt det här felet, vad beror det på? Kan det vara så att det är en extension som vi inte har tänkt på, och då får man helt enkelt uppdatera use caset, skriva ett test case som testar det, sen kan vi fixa buggen och säkerställa att just detta inte kommer tillbaka igen. Det är det jag menar med Test Driven Development men utan Unit Test.

**Intervjuare:**

**När ni nu kör Continuous Integration, är det ett bygge som sker så fort någon checkar in?**

AN:

Varje minut så byggs det automatiskt.

**Intervjuare:**

**Körs det även automatiserade tester då?**

AN:

Vi har precis börjat med det där, så inte i speciellt stor utsträckning. Men tanken är att det ska ske mer. I BRAT har vi en del Unit Test för saker som väldigt ofta orsakar fel, och de körs ju. Men vi ska gå efter att ha bygge, Unit Tester, Ncover. Sen i långa loppen vill vi ha en FXCOP eller något, men det är inte något vi kan föra in nu. Det skulle kräva en massa jobb, då BRAT skulle ge så mycket fel där så. Man får ta det med en regel i taget i så fall, men drömmen är att kunna ta bort Code Review och Guide Lines och ha det i FXCOP alltihopa. Och då gå över till Design Review snarare än Code Reviews.

**Intervjuare:**

**Du menar att en mer erfaren ska gå igenom koden och kanske säga, du skulle kunna gjort det här så här, då hade det blivit bättre.**

AN:

Ja eller att någon frågar jag har tänkt lösa det så här, då kanske han svarar har du tänkt på det här och så vidare. Man ska inte läsa varje kodrad egentligen.

**Intervjuare:**

**Vad har ni för Code Coverage?**

AN:

Vi har inte satt någon ännu. Det är lite sådant jag vill jobba mer med, kvantitativt sätta sådant. I On The Road har de rätt mycket, de ligger på 50% om man kollar rent generellt, och GUI-projekt på 70%, vilket jag tycker är bra för det är så svårt att testa dem om man inte använder något speciellt design pattern för att lösa det där. 100% har jag ingen tanke på men jag vill hellre ha Scrum som fungerar bra så att man kan börja mäta efter ett tag sådant

jag vill veta som linjeförman. Ja visst jag kan veta att vi har 95% Code Coverage men vad kommer det kosta i förhållande till hur många buggar vi släpper?

**Intervjuare:**

**Detaljstyr du inte teamet då? När du säger att du vill ha NCover. Även om inte de känner behovet av det, så går du ut och säger att så ska de göra för att nå Code Coverage målen.**

AN:

Hur jag jobbar i sådana fall är att jag visar dem någonting och ber dem prova det och sedan säga vad de tycker. T.ex. så blev jag introducerad till presenter först UI-pattern. Jag visade det för teamet, de testade det och sedan frågar man dem om varför de vill eller inte vill använda det, och då vill jag gärna ha en förklaring.

**Intervjuare:**

**Har ni haft problem i just att ni inte kan Unit Testa gränssnitt?**

AN:

Nej, börjar vi med Unit Test så har vi så mycket mer att göra än bara det. Men vi gjorde en liten mini applikation där vi körde presenter först.

**Intervjuare:**

**Presenter först är det ett pattern i sig eller?**

AN:

Nej det är ett UI pattern som man sedan slänger in TDD-approach på.

**Intervjuare:**

**Då har vi inga mer frågor.**

AN:

Största skillnaden är alltså att teamet tar ett helt annat ansvar.

**Intervjuare:**

**Som du har upplevt det, eller som de själva upplever det tror du?**

AN:

Det tror jag men det är inget som man enbart ska tillskriva Scrum, utan det är kanske också snarare organisationen. Det var ganska stökigt i början men Scrum har nog gjort att det gått snabbare att nå där vi är idag det tror jag.

**Intervjuare:**

**Finner du det någonsin frustrerande att du inte kan detaljstyra?**

AN:

Nej jag tycker det är jätteskönt att slippa det!

**Intervjuare:**

**Var det teamet som kom med kraven att använda Continuous Integration?**

AN:

Det är Dick som har fört in det, för han tyckte det skulle vara bra. Sen har jag ivrigt påhejat det så att säga. Han gillar nya saker, vet ni vad Hibernate är för något? On The Road kör numera med NHibernate och ett annat projekt också. Det hade jag ingenting att göra med överhuvudtaget. Utan jag tog med dem till en MSDN-live session där de pratade om det, och de nappade på det där då och drog igång det själva.

**Intervjuare:**

**Trevligt, låter som min hemmamiljö, sitter i NHibernate 5 timmar om dagen.**

AN:

Just nu är det ju faktiskt så att vår ScrumMaster arbetar mest med att driva igenom Scrum hos medvetandet hos de andra, även om han också är utvecklare i teamet.

**Intervjuare:**

**Utvecklar Product Owner också?**

AN:

Nej inte i dagsläget, han har ju tidigare varit projektledare för produkten.

**Intervjuare:**

**Ok, tack så mycket!**

AN:

Tack själv.

## **Bilaga 11, Lars Ahlkvist, ScrumMaster**

**Intervjuare:**

**Jaha, då var det intervju med ScrumMaster på Sony Ericsson den 22:e januari klockan 15:00, det är en ganska öppen intervju och vi har lite grundfrågor. Tack för att du tar dig tid. Om vi kunde börja med lite information om dig själv namn och tid i organisationen..**

LA:

Jag heter Lars Ahlkvist och jag har jobbat på Sony Ericsson sen 2003.

**Intervjuare:**

**Hur lång erfarenhet har du i ditt nuvarande yrke, utvecklare, ScrumMaster?**

LA:

ScrumMaster har jag inte varit så länge, inte mer än 3-4 veckor, som utvecklare är det sen 95.

**Intervjuare:**

**Vad har du för tidigare erfarenheter av systemutvecklingsmetoder, såsom RUP, vattenfall, etc.**

LA:

Det var väl egentligen inte vattenfall jag arbetade med innan. Det var tänkt vattenfall men det ändrades efter hand. Trots att specarna ändrades efter hand så var det ganska tydligt vattenfall

**Intervjuare:**

**Vad har du för tidigare erfarenhet av agila tekniker och projekthanteringsmetoder som Scrum, till exempel Test Driven Development?**

LA:

Ett halvår ungefär.

**Intervjuare:**

**Hur upplever du att dina tidigare erfarenheter inom området påverkar din nuvarande roll? Har du mycket lärdom som du kan dra nytta av?**

LA:

Det största är att allting förändras. Även fast man kör vattenfallsmodellen så förändras det.

**Intervjuare:**

**Mmm. Du sa innan att du var ScrumMaster, då tog du en certifiering?**

LA:

Mmm.

**Intervjuare:**

**På Softhouse fick vi höra, stämmer det?**

LA:

Ja.

**Intervjuare:**

**Vad lärde du dig av den så att säga? Har du haft stor nytta av den?**

LA:

Allt som allt var det mest att vi var på rätt bana, men så är det framförallt små tricks och så för att man ska få produktägaren att engagera sig mer, hur pass viktig Product Backlog är, man får Burndown Charts och att det blir relevant.

**Intervjuare:**

**Använder du till exempel mycket Burndown Charts?**

LA:

Japp, det stämmer.

**Intervjuare:**

**Som jag förstod det från en tidigare intervju med er linjeförman så sitter Product Owner här hos er så att säga.**

LA:

Bredvid mig ja.

**Intervjuare:**

**Det fungerar bra att ha Product Owner så nära? Och att han är en del av erat utvecklingsteam från början som jag förstod det.**

LA:

Både för- och nackdelar.

**Intervjuare:**

**Vad är nackdelarna?**

LA:

Nackdelarna är att han gärna vill lägga sig i, fördelen är att man kan ha en väldigt snabb kommunikation med produktägaren: "Hur vill du egentligen ha det, hur hade du tänkt dig på det här?". Samtidigt så har han lättare att ändra sig under tiden, vilket han inte får.

**Intervjuare:**

**Hur har du märkt att han försöker lägga sig i mer? Har han sagt åt någon person att göra på ett speciellt sätt?**

LA:

Nej, han går till mig som ScrumMaster, som det är tänkt. Vi är hårda på det, så att de andra hänvisar till ScrumMastern.

**Intervjuare:**

**Ni har fått in en rutin i det eller?**

LA:

Vi försöker köra det rätt strikt på Scrum på det sättet att inte störa medlemmarna i teamet.

**Intervjuare:**

**Ja, för det kanske kan bli problematiskt om man sitter så nära varandra?**

LA:

Ja, han kommer gärna med frågor och sånt, samtidigt som teammedlemmarna har lättare att bara hugga produktägaren och säga typ "kolla här, var det så här du menade från början?"

**Intervjuare:**

**Hur kommer teammedlemmarna till dig med frågor? För tanken är väl att du ska utbilda dem inom Scrumprocessen så att säga?**

LA:



Ja. Problemet är att jag är en medlem i ett av teamen också, så jag får frågor inte bara som ScrumMaster, så man får lite för stort inflytande egentligen.

**Intervjuare:**

**Finns det en klar avgränsning om de kommer till dig som gruppledare eller ScrumMaster?**

LA:

Nja..

**Intervjuare:**

**Vad kan det vara för frågor som gäller Scrumprocessen?**

LA:

Ja.. Det är lite hur hinner vi med det här det här och hur ska jag planera för detta för nästa sprint, osv.

**Intervjuare:**

**Lite mer praktiskt.**

LA:

Ja, mycket med praktiska ändamål. Vi är en så pass tät grupp att det mesta löser sig själv.

**Intervjuare:**

**Utbildade du då teamet efter din egen utbildning så att säga, i Scrumprocessen?**

LA:

Ja

**Intervjuare:**

**Var det något motstånd? Var det något problem i samband med den utbildningen? Stötte du på konstiga frågor eller så?**

LA:

Inte från teammedlemmarna, däremot produktägaren var lite skeptisk till att tappa inflytande.

**Intervjuare:**

**Med motiveringen att...?**

LA:

Att det han ska få är det som är värt något i slutändan.

**Intervjuare:**

**Har du mött något mer motstånd, förutom då kanske Product Owner, mot Scrum och agila tekniker, att många av utvecklarna kanske inte tyckte det här var så jättebra, tycker att det är jobbigt med TDD eller så?**

LA:

Nej, alla har varit väldigt öppna för det, framförallt linjeföraren, men även organisationen utanför, vilket jag tror är ett måste. Om man kan få med linjen på det hela kan man få det att funka.

**Intervjuare:**

**Vad upplever du personligen är fördelarna med Scrum, med din roll som ScrumMaster, respektive nackdelarna? Har du svårt att hålla isär dina intressen som utvecklare och ScrumMaster?**

LA:

Det kan ju bli konflikter, när man är ScrumMaster och Teammedlem samtidigt, och jag försöker så mycket som möjligt gå tillbaka och hålla mig till ScrumMaster så att teamet får bestämma utan mig. Jag tar rollen som ScrumMaster före rollen som teammedlem.

**Intervjuare:**

**Men det finns ett annat team där du bara är ScrumMaster, och inte medlem? Finns där några andra nackdelar som är specifika till just ScrumMasters roll?**

LA:

Man har inte lika mycket insyn i gruppen. När man är inblandad kan morgonmötena bli ”ja, jag vet vad ni håller på med”, och de andra vet det också. Men i det team där man inte är medlem också har man ibland inte den blekaste vad de håller på med just nu.

**Intervjuare:  
Men ni kör Daily Scrum, oavsett vad?**

LA:

Japp.

**Intervjuare:  
Och det är du som håller i det?**

LA:

Precis.

**Intervjuare:  
Hur satte ni ihop Scrumteamen från början? Var det bara av naturen eller?**

LA:

Linjechefen har satt ihop dem, jag har haft en liten del i det, och försökt att få två liknande team, så att det inte blir ett a- och ett b-lag. Det ska inte spela någon roll vilket team som arbetar med en viss produkt. De ska även ha samma kunskaper så att man sprider kunskapen inom gruppen.

**Intervjuare:  
Eran linjechef sa innan att teamet organiserats så att det fick vissa roller, att någon var bra på någonting och att han fick en lite mer tydlig roll där, har du också upplevt det så?**

LA:

Ja, fast det skapar de själv, så det har de löst själva. Det är bättre, med detta kommer mer information till de andra, och de försöker lära dem och hjälpa dem.

**Intervjuare:  
Så kunskapsspridningen är något som teamet jobbar på även om de har experter?**

LA:

Precis.

**Intervjuare:  
Men de har inte bett dig om råd vad gäller deras interna roller?**

LA:

Nej.

**Intervjuare:  
Hur ofta förändras sammansättningen av teamen?**

LA:

Nu har vi precis gjort en split på det ursprungliga teamet, att dela upp det egentligen på mitten och få in de andra i teamen, egentligen en uppskalning av Scrum som man ska göra.

**Intervjuare:  
Hur har det gått? Har det blivit några konflikter?**

LA:

Nej, inte än så länge. Det var snarare så att jag som ScrumMaster kunde känna att nu var det här teamet tight, alla vet vad alla gör och de jobbar bra tillsammans, lite trist att dela ett sånt team, samtidigt, så ska man få in de andra så kan man inte bara sätta en ny grupp vid sidan om. För kunskapsspridningen så måste man få in dem mer.

**Intervjuare:**

**Hur upplever du att det fungerar att sitta med i en sån öppen arbetsmiljö, med två team, att man hör varandras team när man pratar?**

LA:

Det är inte så farligt. Vi ska flytta om till helgen faktiskt, så att vi sitter mer team-mässigt i fyrklövers-former, för att sitta nära varandra så att man kan sköta de här kommunikationerna.

**Intervjuare:**

**Ber någon i ett team någonsin det andra teamet om råd?**

LA:

Ja. Framförallt på Daily Scrum på morgonen så är det någon som tar upp någonting som jag vet att det andra teamet har tittat på så säger jag till dem att gå och fråga. Allting är ju för produkternas bästa.

**Intervjuare:**

**Vad upplever du är det svåraste för teamen hittills? Har något varit svårt?**

LA:

Lite självstyret. Vill gärna att ScrumMaster ska säga till dem vad de ska göra, och sen att ha ansvaret för att grejerna blir gjorda. De känner inte det här, att någon har sagt till dem, och då måste det blir gjort.

**Intervjuare:**

**Är det något som ändrats med tiden?**

LA:

Ja, de har blivit bättre och bättre på ansvarstagandet.

**Intervjuare:**

**Har du gjort någonting för att upplysa dem och påverka dem?**

LA:

Ja, framförallt på sprintplaneringen så är det dem som bestämmer tiderna och så få som möjligt ska påverka dem. T.ex. jag säger till exempel till dem om de tror på sina tider, och att de efter sprintplaneringen ska kunna följa sina tider också. Annars får vi ju ändra på någon tid innan vi går härifrån. När vi har sagt något får vi stå för det.

**Intervjuare:**

**Lite mer engagemang alltså?**

LA:

Ja, precis.

**Intervjuare:**

**Kan du ge några exempel på hur teamet själva löst konflikter utan att ta hjälp av dig som ScrumMaster?**

LA:

Nej, några konflikter har vi nog aldrig haft.

**Intervjuare:**

**Som vi förstått det är du den mest erfarna utvecklaren här, har det varit några problem i och med att du är ScrumMaster och utvecklare, att de förlitar sig på dig eftersom du är en väldigt erfaren utvecklare?**

LA:

Ja, de kommer ju och frågar, och då är frågan om de kommer till mig som teammedlem eller som chef eller ScrumMaster, den avvägningen är svår. Men mitt eget team, där jag sitter med, frågar mer än det andra teamet gör, så att splitten har gjort att de mer känner att de som ett team ska fixa det själva.

**Intervjuare:**

**Hur många är de i det teamet där du inte medverkar?**

LA:

Fem.

**Intervjuare:**

**Finns det några problem man kan tillskriva Scrumprocessen? Är de dåliga på att fylla i sprint Backlog?**

LA:

Nej, jag har blåslampa. Skämt åsido, så är det mest produktägaren. Vi har pratat mycket om det, och det börjar bli bättre, men han måste ha en full Product Backlog, och den måste vara prioriterad. Han kan ju inte lägga till grejer som vi ska titta på. Vi ska faktiskt lägga en viss tid av sprinten på att estimeras Product Backlog och då måste där vara grejer också.

**Intervjuare:**

**Vad gäller självbestämmande, har du några exempel på beslut som de fattat som en grupp utan att blanda in dig?**

LA:

De vill gärna rådfråga mig, men det är de som bestämmer, så det är ju väldigt mycket upp till ScrumMastern, att om han säger något så kommer de nog att lyssna, iallafall här. Men där är det upp till mig att jag måste ta steget tillbaka, "det är inte jag som bestämmer, det är ni som bestämmer själva". Till exempel nu efter sprintplaneringen så ville produktägaren byta ut en item mot en annan item, så jag hänvisade till teamet, "du får fråga dem, går de med på det så kan vi göra det, säger de nej så är det nej", och då har vi alternativet att vi får planera om igen, eller så kör vi på det vi hade.

**Intervjuare:**

**Har ni terminerat någon sprint?**

LA:

Nej, det har inte gått så långt att någon behövts termineras.

**Intervjuare:**

**Men ni har rätt att göra det.**

LA:

Ja, den rätten har vi.

**Intervjuare:**

**Har du sett några skillnader i teamets sätt att arbeta som du tror att man kan tillskriva Scrum?**

LA:

Ja, man ser ett slut på saker och ting, som när man har en deadline, så har man ambitionen att bli färdiga till dess, om man bara har ett löpande projekt så är det mer "jaja, om detta tar två veckor eller en gör ju inte så mycket". Men när vi nu efter två veckor ska ha någonting att visa.. Just det här att ha något att visa har en stor effekt, man vill ju visa något som är bra.

**Intervjuare:**

**Blir det en starkare gruppsammanhållning på grund av det?**

LA:

Ja man hjälper ju varandra, och det märkte vi framförallt på första sprinten när man började att verkligen hjälpa varandra, och på retrospective tog de upp det, att här hjälpte vi verkligen varandra. Initialt tog det lite längre tid,

men i slutändan så gick det fortare för helt plötsligt hade vi en bättre lösning, och det gick fortare att implementera.

**Intervjuare:**

**Upplever ni att effektiviteten i teamet har förändrats ju fler sprintar ni har genomfört, eller har det varit en ganska liknande kurva?**

LA:

Nej det har blivit bättre och bättre tills vi splittade nu, då gick det ju ner kraftigt, och sen kommer den att bli bättre och bättre igen. Det är ju trots allt fyra medlemmar som aldrig har jobbat i det här projektet, så att det tar ju tid att lära dem projektet också, produkten, vad kan den här produkten, och koden helt enkelt.

**Intervjuare:**

**Vad ser du som dina huvuduppgifter?**

LA:

Huvuduppgifterna är ju att se till att de andra kan utföra sitt jobb. Jag är ju utvecklare dessutom, så jag känner mig delaktig i att se till att allt kommer i mål.

**Intervjuare:**

**När känner du att det är jobbigt att vara i ScrumMaster-rollen på grund av att du kanske vill styra? Kanske för att de inte går tillräckligt fort framåt, har det varit några såna problem?**

LA:

Ja, det kan ju vara så att man tycker att man har en bra lösning, att man inte ska prångla den på, utan låta dem själva komma med en lösning. Eftersom jag jobbat länge med produkten så vet jag ju mycket om den. Och där är det viktigt att skilja på, om de kommer med en fråga "hur hade du gjort" osv., då är det ok att lägga fram "så här hade jag nog gjort det", men jag ska inte gå in i teamet och säga "gör så här", då funkar det inte.

**Intervjuare:**

**Tror du att du hade klarat av det lika bra utan utbildningen i Scrum?**

LA:

Nej. De påpekade många gånger: "ni har inget att bestämma om".

**Intervjuare:**

**Du sa tidigare att Product Owner försökt komma med visst detaljstyre till gruppen, är det likadant med resten av organisationen, att det kommer påtryckningar utifrån?**

LA:

Nej, det har jag inte känt av. Eller till viss del kommer det utifrån, men vi hänvisar hela tiden till Product Owner, vi tar ingenting som är utanför, utan vi har bara en kanal ut, och det är Product Owner. Så om någon kommer till mig så hänvisar jag till Product Owner, kommer de till teamet hänvisar de till antingen mig eller Product Owner, och jag hänvisar vidare. Däremot så har vi kommunikation med användare osv. för att lösa uppgifter, då kan vi gå till Product Owner och fråga "vem är det som bestämmer hur det här ska fungera?", det kan vara en löst specad grej, då säger han att på just den avdelningen fungerar det så här, då kan vi prata med den avdelningen, gör han kanske inte har alla detaljerna. Så det gör vi, men då är det fortfarande ett sätt som gör att teamet bestämmer.

**Intervjuare:**

**Så för att sammanfatta, så är det största problemet för dig som ScrumMaster att hålla isär dina två roller som utvecklare och ScrumMaster, samt att Product Owner inte helt accepterat sin roll?**

LA:

Mmm. Och jag tror att ju längre du har varit project manager, desto svårare är det att bli Product Owner.

**Intervjuare:**

**Hade du haft någon erfarenhet av att vara project manager innan detta?**

LA:

Nej, jag har inte det, men vår Product Owner var ju project manager innan vi började med Scrum.

**Intervjuare:**

**Så du hade kanske fördelen att gå från att vara utvecklare direkt till ScrumMaster?**

LA:

Ja det ser jag inte som en nackdel, däremot att komma från en bestämmande roll till att gå ner till Product Owner eller ScrumMaster tror jag är svårare. Man vill säga ifrån när man är van vid att säga ifrån.

**Intervjuare:**

**Vad håller ni för möten och så?**

LA:

Jag håller i Daily Scrum, och sen har vi Review, och Retrospective, och Sprint Planning, min bit, tillsammans med Product Owner.

**Intervjuare:**

**Har ni förändrat ert arbete efter sprint retrospective?**

LA:

Japp.

**Intervjuare:**

**Många gånger, eller vid varje sprint?**

LA:

Varje sprint har vi retrospective.

**Intervjuare:**

**Men har ni ändringar efter varje?**

LA:

Vi försöker göra små ändringar hela tiden, det går inte att säga "nu ändrar vi allt" för då slår man ju sönder hela arbetssättet som man byggt upp, framförallt försöker vi lyfta upp såna saker där vi gör något bra, så att vi kan fortsätta med det. Och om det kommit in något som var mindre bra så försöker vi sluta med det.

**Intervjuare:**

**Kan du ge exempel på saker som varit mindre bra respektive bra?**

LA:

Mindre bra var en person som började mitt i en sprint, han började sin anställning då. Det tog tid att få upp hans arbetsmiljö, och det är ju omöjligt att planera in sånt i en sprint, det kan ta en dag eller tre dagar innan allting funkar. Nånting som är bra är samarbetet, att alla kan ställa sig bakom någon som har problem så att vi kommer vidare, det har de tagit upp att de tyckt är bra.

**Intervjuare:**

**Vad använder ni för agila tekniker, som t.ex. TDD idag? Använder ni till exempel automatiserade tester?**

LA:

Unit testing har vi börjat använda, men det är ju svårt på ett befintligt system. Så unit testing är det mesta vi kommit fram till, och satsar på framåt. Med system som bygger varje dag och Cruise Control, och vi har en skärm där ute visar att nu checkade vi in något som inte funkar.

**Intervjuare:**

**Har du märkt något motstånd eller någon problematik hos Scrumteamet när det gäller att arbeta enligt det agila? Det krävs ju ändå en viss disciplin. Vi fick intrycket innan av linjefefen att det var lite Hawaii innan Scrum kom in i bilden.**

LA:

Ja, och då hade man ju inget slut heller, om det tog lång tid spelade det ingen roll. När vi började att använda Scrum, dels så var det ingen som var helt införstådd med vad det innebar att ta ansvar för sprinten och all allting blev färdigt, så då var det ju egentligen upp till ScrumMastern, men nu är det mer upp till alla att det blir färdigt.

**Intervjuare:**

**Hur har det gått när sprintarna dragit ut, och man inte blivit klar?**

LA:

Ja det var ju mest när man inte tog sitt ansvar, och "vi gör väl bättre nästa gång".

**Intervjuare:**

**Vad tror du bidragit till att ni tar ert ansvar nu?**

LA:

För att jag sa till dem varje gång att "det är ni som har sagt att det här tar så lång tid, gör det inte det får ni förklara varför också, det kan vara att man planerat dåligt, men då får vi se till att göra det bättre nästa gång.

**Intervjuare:**

**Så du förlitar dig lite på människans vilja att lita på varandra.**

LA:

Ja, annars funkar det inte, man måste lita på varandra, det bygger det på.

**Intervjuare:**

**Hur stora Scrumgrupper har ni nu?**

LA:

Fem, vi hade en grupp på sju innan.

**Intervjuare:**

**Är det någon stor skillnad på att vara sju och fem?**

LA:

Morgonmötena går fortare på fem, och inte bara att det går två personer fortare, för det går mycket fortare. Nu har vi morgonmöten på 5-6 minuter, så pass bra disciplin är det: "det här gjorde jag, det här ska jag göra, detta är problem".

**Intervjuare:**

**Hur var skillnaden med sju?**

LA:

Det var mer försök till problemlösning. När vi började med Scrum så hade vi morgonmöten på typ 35 minuter eller något sådant, och när vi kommit igång och jag blivit ScrumMaster så var vi nere på en 10 minuter, och nu är vi nere på en kanske 6 minuter.

**Intervjuare:**

**Vad händer om inte du är här?**

LA:

Då håller de mötet själva. De organiserar det själva.

**Intervjuare:**

**Ja, det var väl i stort sett de frågor vi hade då. Tack så hemskt mycket!**

LA:

Tack själva.

## **Bilaga 12, Niklas Holmberg, Product Owner**

**Intervjuare:**

**Intervju med Product Owner på Sony Ericsson Niklas Holmberg, den 22:e januari klockan 15:40 2008. Om du kan dra snabbt lite bakgrund om dig, tid i organisationen, roll, och så?**

NH:

Ja, jag jobbar som konsult, först 2004 för cybercom group, jobbade på systemtest, startade sedan utveckling av brat-projektet tillsammans med Anders Nyberg och Gert Olofsson som ett internt projekt på testavdelningen, jobbade med det runt ett år tills vi fick klartecken att det skulle bli ett globalt verktyg och då jobbade jag som utvecklare och testare så jag kom in som testare, och började använda verktyg för automatiserad testning baserat på manuella testfall, sen gått över till att utveckla skript, och nu i april 2007 blev jag anställd på Sony Ericsson, gick in i rollen som projektledare under sommaren någon gång och sen som Product Owner när vi började Scrum i oktober tror jag. Bakgrund som interaktionsdesigner, utbildad på Malmö högskola och lite språkkunskaper sen lunds universitet.

**Intervjuare:**

**Vad står brat för?**

NH:

Batch Remote Automated Testing. Det är en sanning med modifikationer det här namnet då tanken från början var att man på avstånd skulle kunna testa telefoner, alltså att vi har en testmiljö någonstans i världen så ska man kunna logga upp mot telefonerna och testa dem. Det har inte infunnit sig utan det sker lokalt istället så det här Batch Remote stämmer inte riktigt.

**Intervjuare:**

**Var används det? Är det i hela organisationen eller bara Sverige eller Lund?**

NH:

Det används av alla global test centers. Nu vet jag inte hur mycket man får säga, men alla testavdelningar som har en anledning att använda testautomation har det som det officiella verktyget för testning. Sen finns det ju en massa andra verktyg som man kan använda som kanske fungerar bättre i sitt sammanhang, som mjukvarutestning.

**Intervjuare:**

**Har du några tidigare erfarenheter av Scrum?**

NH:

Inga som helst.

**Intervjuare:**

**Agilitet?**

NH:

Nej, det kan jag inte påstå.

**Intervjuare:**

**Vad har du för tidigare erfarenheter av att arbeta deltagande i ett systemutvecklingsprojekt? Du pratade innan om att du var utvecklare också. Men det kanske inte var någon speciell metod?**

NH:

Nej, det var ju "on demand" så att säga, då jobbade vi som en del av testavdelningen som vi tillhörde innan och då fick vi issues så att säga, vi har ett felhanteringssystem som vi idag plockar de svåraste felen från. Så det var väldigt basic, det var inte så mycket ledning i sig. Samtidigt som vi utvecklade projektet och systemet, så var det



lika mycket för vår egen del vi fixade grejer som det var för en specifik kund. Det var en ganska omogen process i sig, jag vet inte riktigt vad man ska kalla den.

**Intervjuare:**

**Vad har du för utbildning om Scrum och dess arbetsprocesser, och vem har givit den?**

NH:

Den utbildningen har skett mest av Anders Nyberg som är sektionschef och det är egentligen bara via samtal faktiskt. Jag har inte läst på dokumentationen, och sen när Lars Ahlkvist hade gått sin ScrumMaster-kurs så tog han upp alla skillnader som han hade lärt sig, så jag har tagit på mig de förändringarna också. Så vi jobbar lite ad hoc.

**Intervjuare:**

**Så det är en mer informell utbildning då?**

NH:

Ja, absolut.

**Intervjuare:**

**Sen har vi arbetet med Scrum team, hur upplever du att det fungerar idag?**

NH:

Det fungerar faktiskt mycket bättre än vad jag trodde från början, och det är lite att man är skeptisk som projektledare för produktägare, då man innan hade allt ansvar, att delegera allting och se till att allting blir gjort och man känner att man har hela ansvaret på sina axlar och om jag inte piskade på så skulle ingenting hända. Har man det synsättet som jag tror många har idag, och många utvecklare också jobbar efter, så är man rädd för att släppa på allting och låta utvecklarna själva bestämma exempelvis hur lång tid saker tar att fixa, som på ett Planning Meeting. Men efter två-tre sprintar så har alla fått väldigt bra ansvarstagande, det är den största skillnaden som jag ser det, det är mycket större ansvarstagande från teamen själva, att de vågar ta för sig, vågar estimeras, och de vågar säga till om de inte skulle hinna med sin deadline, ScrumMastern ligger på också, men jag känner att det totala ansvaret har ökat mycket.

**Intervjuare:**

**Har du förändrat ditt sätt att arbeta allteftersom?**

NH:

Ja, jag springer ju inte runt och ifrågasätter vad alla gör hela tiden längre, man kan ju säga att projektledarrollen har flyttats från mig till ScrumMastern mycket mer, även om ScrumMaster inte heller ska vara piska, så ska ju han se till att saker blir gjorda, och se till att inte saker faller mellan stolarna, och rapportera det till mig som ägare. Så han har mer den operativa rollen som jag hade tidigare, medan jag bara väntar på releaser och resultat, planerar backloggar och så. Nu förutsätter jag att det fungerar som det ska mycket mer än jag gjorde tidigare.

**Intervjuare:**

**Hur beskriver du din relation gentemot ScrumMastern? När går du till honom?**

NH:

Gentemot tidigare eller?

**Intervjuare:**

**Nej men rent generellt.**

NH:

Ja, den är ganska, iallafall med Lars som är väldigt ansvarstagande, han gör ju livet mycket lättare för mig som produktägare genom att jag kan täcka av min roll mer, så jag kan jobba mer på kundrelaterade frågor, och jobbar med att sälja in produkten mycket mer, och se till att vi är på rätt mål med allting, kravinsamlingar och liknande, så jag tycker att han gör ett jättebra jobb, och har mycket stor förståelse för projektet.

**Intervjuare:**

**Hur hanterar ni förändringar av krav idag? Vem kommer med dem?**

NH:

Där är lite problematik idag, för det finns inte några direkta externa krav, så kravhanteringen är något vi driver själva mycket mer än vad en egentlig kund borde driva till oss. Så vi är lite på två stolar egentligen kan jag säga.

**Intervjuare:**

**Vilka får ni mer in krav från, användare?**

NH:

Det är folk från testavdelningar som behöver nya möjligheter att testa telefoner.

**Intervjuare:**

**Har det blivit några svårigheter med prioritering?**

NH:

Nej, tvärtom, det har känts som att vi kunnat fokusera mycket mer, vi är mycket mer villiga än tidigare att få in krav, för då vet vi att vi utvecklar någonting som får användning, det är ju mycket svårt att gissa själva.

**Intervjuare:**

**Hur prioriterar ni de krav som kommer in?**

NH:

Vi prioriterar dem efter hur många användare det är som blir påverkade, vi har kartläggning nästa månad över alla användare så att vi ska kunna se vad olika features påverkar och om det skulle ligga några [ohörbart] Så just nu tar vi lite av varje och ser till att blidka våra kunder på alla nivåer så att ingen blir utanför, så det blir lite av varje.

**Intervjuare:**

**Ser du det som ett problem att "beställaren" är inom den egna organisationen?**

NH:

Ja, för ofta vet inte beställaren att den är en beställare, och det blir det klassiska att saker och ting faller mellan stolarna. "Vi trodde att ni skulle" eller så. Är man en riktig kund så har man ju ett stort fokus på att man ska ju driva in det man har betalt för så att säga, nu är det ingen som betalar egentligen.

**Intervjuare:**

**Hur presenterar ni det idag?**

NH:

Vi bara gör releaser, och skickar ut information om det som finns tillgängligt, från och med nästa sprint så ska vi bjuda in alla intressenter till våra retrospective-möten, så att de får vara med och avgöra om det är bra eller inte. För hittills har vi bara suttit själva. För egentligen är det ju våra kunder som ska göra ett godkännande på om deras issue har blivit implementerat som det ska.

**Intervjuare:**

**Du menar Sprint Review Meeting?**

NH:

Ja, såklart, så heter det ju! Jag är ju inte med på retrospective, de bara informerar mig, men det hjälper ju inte så mycket, det är ju slutkunden som har ställt kraven från början, så de ska få säga sitt också.

**Intervjuare:**

**Har ni märkt att några krav varit helt uppåt väggarna, och hur hanterar ni sånt idag?**

NH:

Ja, vi implementerar ju inte, vi lägger ju inte något i backloggen som vi inte har för avsikt att genomföra utan det är en dialog, det är många som ställer krav på saker som inte går att uppfylla, vi vill till exempel ha ett verktyg som alla andra har, så då får man ta en diskussion, vi har alltid med nya kunder ett möte, där vi ber om use-cases

och business-case för de man vill ha gjort, sen får vi värdera om det är någonting som vi kan lägga in. Vi kan inte tillfredställa alla små önsknigar, utan det får bli efter vad som går att göra.

**Intervjuare:**

**Skulle du vilja ändra sättet ni arbetar idag?**

NH:

Nej, det här känns väldigt bra hitintills, i våra roller så är vi fortfarande ganska glada allihopa, och det funkar. Det kanske blir konflikter längre fram, för vi jobbar snabbt, men nu känns det bra, absolut.

**Intervjuare:**

**Men du är bara Product Owner för det ena teamet?**

NH:

Nja, det är min produkt så jag äger hela produkten för båda teamen, men vi har ju fler produkter än bara brat, och det ena teamet kommer eventuellt att arbeta med andra saker, just nu sitter de med brat men kan arbeta med andra verktyg i senare sprintar. Men jag är bara Product Owner för brat.

**Intervjuare:**

**Har det varit problem att vara i två grupper?**

NH:

Nej, vi har ju precis börjat så jag vet inte.

**Intervjuare:**

**Vad ser du för nackdelar, respektive fördelar med Scrum?**

NH:

Ja, fördelarna tycker jag ju som sagt är större ansvarstagande, mer koll på var tiden går, det känns som att jag vet vad alla gör, och kan ju hela tiden följa utvecklingen via Burndown Charts osv. Nackdelen är väl att man fortfarande inte vet hur mycket man tjänar på det, det går inte riktigt att säga, för det är ju väldigt mycket möten, och mycket smågrejer som ska till, som planning, retrospective, det är den här points-beräkningen av hur svårt det är att utveckla sina tasks, så skulle man ha teamen med riktigt vassa utvecklare som var självgående och inte kört Scrum så kanske man skulle göra samma saker fast snabbare, men det vet man ju inte. Vi får se lite hur vi ska utvärdera i slutändan, men just nu känns det bra, och om man tänker tillbaka lite på hur det var tidigare så var vi ju inte supererfarna utvecklare, men jag tycker att det går mycket bättre nu.

**Intervjuare:**

**Upplever du att det är stor skillnad på nivån mellan utvecklarna?**

NH:

Ja, en del, både av erfarenhetsskal, personlighet, det kan vara såna saker som att man inte arbetat med verktyget innan, eller en sprint log. Vissa är ganska nya, så det är klart det är olika nivåer.

**Intervjuare:**

**Påverkar du teamet under en sprint på något sätt?**

NH:

Nja, jag kan ju påverka sprinten, jag kan säga om vi får in någonting: ”nej, det här går inte, vi har fått in jättemånga nya akuta issues som vi måste fixa”, då kan vi avsluta sprinten och starta en ny i så fall, och vi får göra en ny bedömning av hur vi ska kunna följa den.

**Intervjuare:**

**Har ni gjort det hittills, avbrutit?**

NH:

Nej, vi har en gång efter ett Sprint Planning Meeting senare samma dag haft ett nytt möte där vi omplanterat, och vi har tagit bort saker när vi inte har hunnit liksom.

**Intervjuare:**

**Hur upplever du att teamet har förändrat sitt arbete, eller upplever du någon förändring?**

NH:

Ja, jag tycker att de flesta har känts tryggare i sin roll, jobbar mycket mer utan att fråga en central person som brukade vara med innan, speciellt frågar man varandra mycket mer inom teamet, öppnare kommunikation.

**Intervjuare:**

**Effektivitetsmässigt, tror du att det blivit bättre?**

NH:

Nja, jag tror det är svårt att räkna på ännu, jag har ju i och för sig bara varit projektledare här sedan i somras, jag vet inte riktigt.

**Intervjuare:**

**Med bakgrund som projektledare, blir man frustrerad av att sitta så nära utvecklarna men inte kunna detaljstyra?**

NH:

I början kände jag det, att jag blivit av med all min makt, och all min påverkan, samtidigt så gör det inte så mycket. Den påverkan jag kan göra nu är ganska minimal, det finns bara en viss "timeframe" liksom, hade man jobbat med ännu mer krävande kunder hade det kanske krävt ännu mer skiftningar under varje sprint också kanske det inte gått att arbeta enligt Scrum om man hela tiden måste förändra sina prioriteringar. Nu ligger jag inte på så mycket, vi har inte så mycket att göra så att vi behöver prioritera om våra sprintar.

**Intervjuare:**

**Var det något speciellt som fick dig att minska dina frustrationer, att du kanske inte hade lika stora förväntningar på dina krav att detaljstyra?**

NH:

Ja, resultaten i sig över vad de lyckats göra, det lugnar ju en. Men i början så tyckte man att det tog för mycket tid, det händer ju ingenting, vi har bara möten, och alla ska vara delaktiga, det kändes så onödigt, jag hade ju kunnat bestämma. Hela det tankesättet förändras ju när man inser att det ju går fortare om alla, inte initierat, men kunskapsnivån höjs och alla får vara delaktiga, de klarar att avgöra saker själva utan att man tvingar dem och säger till vad de ska prioritera. Så resultaten av de första tre sprintarna kanske inte var de bästa, men när vi kom till de senare så tyckte jag att det var väldigt bra, och därför känner jag inga frustrationer längre.

**Intervjuare:**

**Du sa att det var mycket möten, upplever du Scrum som en tunggrodd process, att det tar mycket tid, mycket formellt?**

NH:

Ja, mer än tidigare såklart, men då krävde ju teamet mer av mig istället, så jag fick lägga mer av min tid till att detaljstyra utvecklarna istället för att lägga tid på kunderna som egentligen borde driva och komma till projektets nytta. Så det kanske är tungrott i starten av varje sprint, och i slutet av varje sprint så är det lite extra möten, men resten är inte jag med på, t.ex. på morgonmötena, och jag involveras inte i utvecklingen så den vinsten av tid tycker jag är bra som produktägare.

**Intervjuare:**

**Du sitter i samma miljö som utvecklarna och ScrumMaster, upplever du några problem i det?**

NH:

Nej, absolut inte.

**Intervjuare:**

**Så om dom säger något som du tycker är helt befängt..**

NH:

Nej men vi interagerar ju fortfarande, det är inte så att jag är tyst, om jag till exempel blir bjuden till morgonmötet och vill gå så är det klart att jag får prata, det är ok liksom, annars blir det bara en massa statusrapporter och såna grejer som inte har med det att göra. Men annars är vår kommunikationslinje ganska öppen, och jag känner ju inte att jag är överkörd för att jag är produktägare istället för ScrumMaster eller vanlig projektledare. Det är fortfarande samma kommunikationsnivå, men jag detaljstyr ju inte och säger hur de ska fungera och göra sina uppgifter, utan alla följer fortfarande sprintarna och gör sina tasks.

**Intervjuare:**

**Hur tror du det hade blivit om du suttit i ett annat hus här?**

NH:

Då tror jag att jag hade haft mindre insikt såklart i den dagliga verksamheten och haft större krav på vad som levereras. Jag tror också att jag skulle kräva in mer statusrapporter från ScrumMaster, för nu har vi ju mycket omedveten kommunikation, och varje gång man hör någon säga något så stannar man ju upp och bekräftar det och sen så är det klart liksom.

**Intervjuare:**

**Du menar att du skulle få skickat till exempel Burndown Charts på varje sprint?**

NH:

Under sprinten? Nej, vi har ju en monitor där vi har tre stycken Burndown Charts som man kan kasta ett öga på, och jag har stor tillförlit på ScrumMaster, han är ju utbildad och han är väldigt driven för det han gör, så jag vet att han skulle säga till om han inte tyckte att de såg ut att möta slutdatumet ordentligt.

**Intervjuare:**

**Så du tror att den tilltron är rätt viktig?**

NH:

O ja, ScrumMastern och produktägaren är ju som Piff och Puff, de måste ju verkligen sitta ihop. Man kan ju inte slåss om makten, och på något sätt gå mot olika mål.

**Intervjuare:**

**Men du och teamet måste och ha den typen av relation eller?**

NH:

Nja, egentligen inte på samma sätt. Det är ju bäst om alla är vänner, men ScrumMastern blir ju produktägarens förlängda arm i och med att det är mycket av de uppgifterna som projektledaren hade tidigare, speciellt när man varit en teknisk projektledare.

**Intervjuare:**

**På vilket sätt menar du att en teknisk projektledare går ihop med ScrumMastern?**

NH:

Han ser ju till att saker och ting blir gjorda, hos oss är han ofta den personen som många har tilltro till tekniskt, har lösningar, och om han inte har det så frågar han produktägaren direkt: "hur ska vi göra i det här fallet?". Och det är lite som en teknisk projektledare, som stöttepelare när det gäller att leda utvecklingen.

**Intervjuare:**

**Men teamet går direkt till dig också och frågar?**

NH:

Ja, om det till exempel är något som är med GUI'et eller någonting som de har ett krav eller om de inte förstått kravet. Och det händer ju ibland att vi får sätta en investigate på en task för att vi vet inte riktigt och jag är lite osäker från början, så jag vill inte sitta och bestämma att såhär ska det se ut, för det är inte mitt mål liksom, vi har ju en kravspec som vi går till. Men visst kommer de till mig fortfarande, och ofta är det att de säger till ScrumMaster att "vi fattar inte", och eftersom jag sitter bredvid honom blir jag inblandad direkt.

**Intervjuare:**

**Den här transparensen man snackar om i Scrum, att man ska snappa upp grejer som sägs och sådär, har du nytta av den, nu när du sitter här?**

NH:

Ja, absolut.

**Intervjuare:**

**För det är nog ganska ovanligt att produktägaren sitter med själva teamet.**

NH:

Ja, det kanske det är. Men det är ju som du säger, att sitter man inte med så får man ju kräva in det, och då måste man ha en massa modeller för det, för hur jag ska kunna veta om allt vi har i luften, det känns ju lite ångestframkallande att sitta och lägga ner all sin tid på att säkerställa att man alltid har den information som är den senaste.

**Intervjuare:**

**Tidigare har du haft mycket kontroll, i Scrum så har man inte den kontrollen på det som utvecklas dag för dag, du får via Review Meetings reda på statusen, och har de inte gjort det de ska då, då kan du ta och agera då, men nu kan du eftersom du sitter med dem agera mycket tidigare.**

NH:

Ja, så blir det ju, och där kan jag se en nackdel om jag suttit utanför, om du inte kan detaljstyra det så har du ju bara det fönstret som är varannan vecka eller var fjärde vecka eller hur lång en sprint är, och någonting ska in i releasen, det kanske är extra farligt. Egentligen ska man ju hela tiden bli informerad av ScrumMastern var man är nånstans, och vad man har tillgång till, Burndown Charts osv. Men som sagt, jag har inte riktigt behov av den här tvångsmässiga kontrollen eftersom jag sitter med.

**Intervjuare:**

**Så du skulle klara dig lika bra utanför?**

NH:

Nej, återigen, då skulle jag kräva mycket mer av informationsflödet, jag skulle behöva mer definierade processer för vilken information jag kan få ut och ta beslut. Tilliten minskar, men jag vet att informationen på mitt bord minskar om jag inte ser till att jag får den.

**Intervjuare:**

**Vilken information är det du hade tagit in i så fall?**

NH:

Jag skulle behöva tvinga in dagliga rapporter: ”Vad händer, vad har ni lyckats med, vilka risker har vi för tillfället?”, även ”vad kommer ni inte att hinna med?” osv. Kanske en tolkning över vad Burndown Chartet egentligen innebär. Sen är det ju massor med andra saker som jag får reda på och slipper tänka på. Men man måste sätta mer tillit till sin ScrumMaster.

**Intervjuare:**

**Då har vi nog inte fler frågor, tack så hemskt mycket!**

## **Bilaga 13, David Jönsson, medlem i Scrum Team**

**Intervjuare:**

**Intervju på ScrumTeam-medlem David Jönsson den 22:a januari klockan 16.10 på SonyEricsson i Lund.  
Tack för att du tar dig tid.**

DJ:

Varsågod.

**Intervjuare:**

**Om du kan berätta lite om dig själv, tid i organisationen och bakgrund.**

DJ:

Jag heter David Jönsson och jobbar som konsult på SonyEricsson, min roll i är dels att jag är CM för sektionen, sen är jag utvecklare i ett ScrumTeam under sprinterna. Och ScrumMaster för ett Scrumprojekt mot ett projekt mot Kina.

**Intervjuare:**

**Bakgrund?**

DJ:

Tidigare har jag jobbat på ett internationellt mjukvaruföretag i Malmö och de körde inte Scrum utan de körde vanlig traditionell utvecklingsmodell där de hade en inpool med fall, buggar och så vidare, och sen så hade man ett change request möte då där den gruppen jag ingick i (som analyserade fallen) och en arkitekt och kanske någon utvecklare var med. Sen så stoppade man in lämpliga fall i projektet på utvecklarna som då ägde var sin modul av applikationen.

**Intervjuare:**

**Du har ingen tidigare erfarenhet av Scrum?**

DJ:

Nej, ingen alls.

**Intervjuare:**

**Agila tekniker som TDD och så där?**

DJ:

Nej inte..nej.

**Intervjuare:**

**Ja konstig fråga kanske. Hur upplever du att dina tidigare erfarenheter påverkar din nuvarande roll, positivt och negativt. Har du reflekterat över att så här är bra att arbeta, så här är inte bra?**

DJ:

Ja mina tidigare erfarenheter känner jag ger en viss perspektiv för processen, processmässigt. Man har någonting att relatera mot. Ja, det känner jag hjälper så att säga. Som exempel känner jag att Scrum jag arbetar i nu, där märker man skillnad mot tidigare arbetsätt.

**Intervju:**

**Vad har du fått för utbildning av Scrum och dess process?**

DJ:

Ja utbildningen har varit i gruppen att vi har haft lite genomgångar. Anders, sektionensledaren, har gått igenom grunderna i Scrum. Sen har jag läst lite på nätet själv, och vi har pratat i gruppen och delat information med varandra. Så jag har inte läst någon Scrumbok eller gått någon kurs.

**Intervjuare:**  
**ScrumMastern har informerat?**

DJ:  
Ja han har ju nyss gått sin kurs, och han har informerat en del.

**Intervjuare:**  
**Vilka agila tekniker använder ni i ert ScrumTeam idag? Unit Tester osv?**

DJ:  
Ja, vi kör Unit Testing och CI är precis färdig så den är på G den också. Övrigt vet ej, jag är så dålig på allt vad Agile begreppet innefattar så att säga.

**Intervjuare:**  
**Vad är din åsikt om agila tekniker?**

DJ:  
Jag tycker de är intressanta och de är värdefulla. Alltså att testa alla metoder för utveckling tycker jag är viktigt, att ha en dynamisk process. Jag tycker man ska vara positiv till förändring.

**Intervjuare:**  
**Men det är märkbar skillnad mot tidigare processer du arbetat med?**

DJ:  
Ja alltså den är... du menar Scrummetoden?

**Intervjuare:**  
**Ja att använda agila istället för mer fasta?**

DJ:  
Ja, det tycker jag.

**Intervjuare:**  
**Vem har pushat på användandet av detta? CI osv.**

DJ:  
Det är Anders som nog stått för det mesta. Det har ju kommit fram idéer och så från oss, men jag har uppfattat det som att det är han som driver det mesta framåt så.

**Intervjuare:**  
**Kan du ge några exempel på märkbara skillnader i sättet att arbeta i Scrum, och det du gjort tidigare? Positiva och negativa aspekter av det?**

DJ:  
Om man tar det positiva så får man bättre uppfattning om vart man är i utvecklingsprocessen, hur långt man har hunnit. Det ger en då ju bättre bild om att kommer man att hinna så att säga i teamet. Kommunikationen har blivit bättre känns det som. Det känns mer som ett team som jobbar tillsammans och man tar kollektivt ansvar då.

**Intervjuare:**  
**Vem har satt ihop teamet?**

DJ:  
Det är Anders också, jag vet inte om han har sagt det till er? Men jag tror ProductOwner också har påverkat sammansättningen.

**Intervjuare:**  
**Men ni har inte kunnat ge feedback på vilka som ska vara med?**



DJ:

Jo, vi hade kunnat säga till om vi hade velat så att säga. Om vi tyckt det var något som var galet. Jag tror de teamen vi har nu är väldigt genomtänkta beroende på vad vi har för roller i gruppen, bra sammansättning..

**Intervjuare:**

**Du sa roller, har ni naturliga roller i ScrumTeamet, att någon är expert på ett område?**

DJ:

Ja lite vagt, en stor skillnad kanske är också vilka applikationer man arbetat med tidigare främst. Eh.. att man har specialkunskap kanske till viss del. Men det är inte så utmärkande i vårt team, att någon person har specialkunskaper. Jag inriktar oss på att alla ska kunna lösa alla problem. Det är väl lite mer om man arbetat med en viss del av applikationen tidigare som man kanske då lättare gör den saken.

**Intervjuare:**

**Hur går kunskapsspridningen till inom gruppen, utbildar ni varandra?**

DJ:

Ja man kan alltid fråga, sen så kan man få förklarat hur delar fungerar om man vet någon som jobbat med en viss del. Muntlig nivå fungerar det bra på alltså.

**Intervjuare:**

**Ni har ju börjat nyligen med Scrum, var det några svårigheter?**

DJ:

Vi kanske inte visste varför vi gjorde saker, riktigt. Och sen var det ju rätt nytt hela processen och så vidare. Hur man skulle hantera rollen och ens uppgifter. Det var en inkörningsperiod för att lära sig hur det fungerade, tog väl en eller två sprintar innan det kanske började fungera. Vi hade stor hjälp av verktyget Anders hämtade, ScrumWorks som han kanske berättat om. Det tyckte jag var ett bra hjälpmedel det tvingade in en kanske i ett visst sätt att jobba på men just att det gick snabbt att ta åt sig hur det skulle genomföras så att säga. Men just att man lärde sig processen var bra, underlättade en del arbete.

**Intervjuare:**

**Tror du att det hade gått bättre om du hade fått annan utbildning, mer formell?**

DJ:

Ja lite hade det nog hjälpt.

**Intervjuare:**

**Tycker du organisationen, linjeföraren osv., stödjer ert sätt att arbeta, arbetsmiljö etcetera?**

DJ:

Ja det tycker jag.

**Intervjuare:**

**Upplever du det som SM samarbetar bra med er, hur fungerar det mot er utvecklare? Hur styr han processen?**

DJ:

Som utvecklare kollar jag med han hur man ska hantera processen. Han är ändå chef över processen, han vet och kan ge svar. Sen vet jag inte, i rollen som att skydda teamen har jag inte använt honom som det. Men jag tror att det kan fungera där också, att jag då går till honom om någon annan försöker rycka tag i mig under en sprint.

**Intervjuare:**

**Säger han åt er vad ni ska göra, eller upplever du som att han guidar er?**

DJ:

Hur menar du då, säga till processmässigt?

**Intervjuare:**

**Ja eller om arbetet går lite knackigt, eller om ni inte hinner med allt ni ska göra?**

DJ:

Ja alltså på de här Scrum-mötena så nja, sen vi fick en fast SM så har jag jobbat i en egen sprint med Kina så jag kan inte svara på det just nu. Men generellt sätt tidigare så när vi hade flytande ScrumMaster så fungerade det bra.

**Intervjuare:**

**Händer det någon gång när ni diskuterar lösningar att ScrumMaster lägger sig i? Eller går ni och frågar honom när ni kört fast?**

DJ:

Nej, så har jag inte upplevt det. Att vi har frågat honom om det.

**Intervjuare:**

**Ok.**

DJ:

Vi har nog som grupp haft mer kontakt med ProductOwner direkt.

**Intervjuare:**

**Har du upplevt några svårigheter kopplat till samarbetet med ProductOwner. Fungerar det bra att han sitter med er?**

DJ:

Ja ibland har ju kanske ProductOwner saker som han vill att vi ska göra som inte finns i sprinten, då ska man kanske försvara sig mot det, i Scrumprocessen. Men det kan ju vara viktiga saker, hur gör man om man kommer in och upptäcker nya buggar. Då får man ta det med PO och om vi ska omfokusera sprinten och ta in detta i sprinten.

**Intervjuare:**

**Är detta något som har ändrats med tiden? Har samarbetet förändrats?**

DJ:

Det var nog svårare i början, det har förbättrats med tiden. Jag tror det tar tid att bygga upp en förståelse för detta om hur man ska förhålla sig.

**Intervjuare:**

**Tror du att det finns någon som kunde gjort i ett tidigare skede för att underlätta denna förståelse?**

DJ:

Svårt, det är något som får nötas in, det tänket. Det kanske är lättare för andra än för vissa. Jag tycker man har fått lära sig att tänka om, från hur man gjort tidigare. Man får låta det ha sin gång..

**Intervjuare:**

**Hur tycker du att det fungerar att planera en sprint i taget, tycker du det är kort med två veckor?**

DJ:

Jag tycker det är lagom.

**Intervjuare:**

**Vad är de positiva grejerna med att så kort planering ändå?**

DJ:

Ja man har ju kort fokus, det är lätt att hålla reda på och se slutat liksom när saker ska vara klart. Det är mer överskådligt kanske.

**Intervjuare:**  
**Kommer ProductOwner in ofta i en sprint och vill lägga till grejer?**

DJ:  
Nej det gör han inte.

**Intervjuare:**  
**Upplever du att ni efterlever Scrumprocessen, fyller ni backlogen?**

DJ:  
Ja det tycker jag att vi gör.

**Intervjuare:**  
**Vem kontrollerar det?**

DJ:  
Det är väl ScrumMaster som ska göra det, kontrollera processen. Men Backlogen är väl ingen som kontrollerar men det upptäckt vid de här Sprint Planning Meeting, om något inte stämmer.

**Intervjuare:**  
**Har ni haft problem med det?**

DJ:  
Ja till viss del, om jag tänker efter. Det kanske inte alltid en Backlog-item är rätt inlagt, när man synar den. Men det är inget stort problem.

**Intervjuare:**  
**Innehåller PB alla typer av krav, funktionella och icke funktionella, snabb prestanda, funktioner och så vidare?**

DJ:  
Nej inte alla sådana, utan det är DMSer eller mer konkreta väldefinierade grejer som ska fixas.

**Intervjuare:**  
**Vad händer om ni inte klarar av det som ska genomföras i en sprint?**

DJ:  
Då flyttas det till BL och ja kanske kommer med i nästa sprint kanske.

**Intervjuare:**  
**ProductOwner gör det?**

DJ:  
Ja, vi flyttar saker i samförstånd med PO, om man inte hinner med ska man ju prata med honom och komma överens om att det är ok att man inte gör den i sprinten osv.

**Intervjuare:**  
**Har ni blivit bättre på estimerar tycker du?**

DJ:  
Ja det tror jag faktiskt, den estimeringen görs ju i grupp nu och vi har haft rätt bra uppföljning på tidsestimat hur de har förhållit sig till resultatet. Men jag tror vi har blivit bättre på det, att estimerar. På den data jag sett i alla fall.

**Intervjuare:**  
**Det tror du sker via erfarenhet?**

DJ:

Just estimeringen, ja det är väl både alltså. Det är nog mera att man att man verkligen tar sig tid. Att man tar en hel dag för en Sprint Planning Meeting, det har jag inte sett innan, att man tar den tiden.

**Intervjuare:**

**Hur upplever du att organiseringen av arbetet har förändrats efter starten?**

DJ:

Den har nog inte förändrats så mycket tror jag.

**Intervjuare:**

**Men det är ni som sköter det, Team-medlemmarna?**

DJ:

Ja men jag har inte upplevt den förändrats så mycket.

**Intervjuare:**

**Har du exempel på beslut som teamet fattat tillsammans, har ni problem i utvecklings eller så, pratar ni med varandra då?**

DJ:

Ja är det någon i gruppen som har problem, man kanske ska prata med Product Owner. Hur vi ska jobba vidare med det här, hur lösa problemet då. Det var väl ett exempel.

**Intervjuare:**

**Men ni har inte haft konflikter som ni varit tvungna att ta hand om som grupp?**

DJ: Nej inte vad jag.. som jag ser det gruppkonflikter, nej.

**Intervjuare:**

**Innan hade ni inte kört CI så länge, vart det problem kopplade till att inte ha gemensam repository för kod, alltså delad kod?**

DJ:

Jo men vi har ju kört på samma branch men samma kodspår, checkat ut och in till samma repository.

**Intervjuare:**

**Har ni några regler för vad som gäller vad som checkas in?**

DJ:

Hur menar du, alltså, man ska ha kommit till ett tillstånd då man själv testat koden man gjort innan man checkar in. Mer än så har vi väl inte. Men du har tagit det till en viss fall status, det har varit ett fall kanske, då ska det nog klart för testning princip.

**Intervjuare:**

**Så det är vad som definieras som en done?**

DJ:

Implement, done, ja . Jo det tror jag vi säger.

**Intervjuare:**

**Är ni i gruppen homogen grupp, har ni samma bakgrund och värderingar?**

DJ:

Värderingar har vi väl, ja det är nog rätt så lika, annars har vi rätt olika bakgrund gällandes erfarenheter.

**Intervjuare:**

**Ja teknisk expertis, men ni ser på samma sätt hur saker ska göras?**

DJ:

Ja det varierar väl, men vi är bra på komma fram till lösningar tillsammans, men rätt lika är vi.

**Intervjuare:**

**Hur har ni möjlighet att påverka ert arbete efter låt oss säga ni har haft ett Sprint Review Meeting? Och Retrospective. Och ni känner att ni behöver förändra något, vad gör ni då?**

DJ:

Det har ju tagits upp på Retrospective och då tar man väl beslut om man ska göra ändringar i processen och då gör man det.

**Intervjuare:**

**Hur stora ändringar kan ni göra kan ni ändra allt?**

DJ:

Ja det kan vi göra, det tror jag.

**Intervjuare:**

**Har du exempel?**

DJ:

Vi gör så att vi skrev headerhistory på all incheckad kod och nu ska man även skriva fallnummret där också så vi kan spåra det. Det drev jag igenom nyss.

**Intervjuare:**

**Ni sitter i en öppen miljö, ser du några nackdelar respektive fördelar med att ni sitter så öppet?**

DJ:

Du menar i motsatt till kontor?

**Intervjuare:**

**Ja kontor, eller om ni satt längre bort ifrån varandra.**

DJ:

Ja kontorslandskapet är mer socialt, helt klart!

**Intervjuare:**

**Ingen har motsatt sig att sitta i den typen av miljö. Upplever du att det är lättare att följa andras progress, att man snappar upp saker?**

DJ:

Man kan ju höra andra saker så visst. Overall progress så tittar man hellre på verktygen som vi har.

**Intervjuare:**

**Ja, ska vi se här. Har du sett några skillnader i teamet sätt att arbeta som du vill tillskrida Scrum?**

DJ:

Bra fråga. Det var det jag sa innan, att jag tycker kommunikationen nog har ökat, och sen har grupptänkandet som team kommit. Ehh det är lite bättre, det är mer kontroll i Scrum, man kan följa upp och se hur saker är gjorda osv. så att man kanske bättre följer upp saker och ting.

**Intervjuare:**

**Mmm är det något du vill ändra i processen, Scrumprocessen ni använder. Tycker du det fungerar bra som ni har det nu?**

DJ:

Jag tycker det fungerar bra som det fungerar nu men skulle jag upptäcka något är det fritt fram för mig att pusha fram och ändra. Men inget jag stört mig på processmässigt som jag kommer på just nu.

**Intervjuare:**

**Hur är det att arbeta utan att få detaljstyrning utan att ni nu löser det i grupp, att ta allt ansvar själv?**

DJ:

Ja det är intressant faktiskt, att man inte har detalj styrning, att det är kollektivt ansvar. Det har ju... hm.. jag kommer inte ihåg problemet vi hade...

**Intervjuare:**

**Men hade det med ansvarstagande att göra?**

DJ:

Ja just det. Jo, ibland i början kändes det som att det verkade liksom att någon skulle vara stående kontaktperson utåt, att en i gruppen då var kontaktperson gentemot Product Owner, och det diskuterade vi och då kände jag lite som att då började bli att den personen skulle få (även om det var roterande) så skulle den personen agera lite mer som en projektledare, nej inte projektledare vad heter det. Lite över de andra rollerna och kanske då information hiding dyka upp. Då diskuterade vi hur självledande gruppen skulle vara, men det slutade med att vi inte hade några sådana personer.

**Intervjuare:**

**Upplever du gruppen som självorganiserande, och självstyrande?**

DJ:

Jag upplever det så.

**Intervjuare:**

**Var det så från början, eller har det förändrats efter sprintarnas gång?**

DJ:

Ja, egentligen vill jag minnas att vi varit självorganiserande. Men blivit bättre på organisera oss eller vad man säger, förbättrat kommunikationen, ju längre vi arbetat. Men jag tycker vi varit självständiga och fungerat som team från början, gick ganska snabbt att komma in i det. Men visst det kollektiva ansvaret har nog förbättrats.

**Intervjuare:**

**Mmm de flesta punkter har vi nu gått igenom nu. Vi har tagit upp dem under diskussionen.**

DJ:

Jag kan säga angående min distans sprint mot Kina så upplever jag det väldigt positivt. Att det i och med vårt Scrumverktyg att det är lätt att följa upp vad de gör på distans och se hur långt vi har kommit. Jag tror det hade varit svårare om det inte funnits något verktyg och tydlig process. Så då känns det rätt nyttigt även om det fortfarande är lite problem med tidsskillnader. Men det känns som om Scrum underlättar för mig att arbeta!

**Intervjuare:**

**Vem är ProductOwner för dem?**

DJ:

Det är samma som för de andra sprint-teamen.

**Intervjuare:**

**Hur går Sprint Planning Meeting till?**

DJ:

Som vi gör nu... Alltså vi har tidsfönster på två timmar vilket är för lite för ha ett Sprint Planning Meeting så vi diskuterar just nu lite hur vi ska lösa det, men som det ser ut nu så kommer de göra en analys och sen kommer jag göra en analys sen får vi snacka om det vi inte tycker är lika. Ehh. Det bästa vore ju om man kunde ha ett längre möte så att säga, kanske via telefonkonferens eller något, men jag vet inte. De gillar inte det riktigt, att prata telefon, tala engelska är ett problem tycker de. Vi kör ju Scrummöte varje dag och där räcker ju tidsfönstret, över chat som Messenger och det fungerar rätt bra för den typen av möte.

**Intervjuare:**

**Spännande, inte så vanligt med Scrum på distans. Finns en artikel på INFOQ där de diskuterar det distans, Scrum på någon Screencast eller vad det heter. Ett tips att kolla in kanske?**

DJ:

Jag läste en annan artikel som Lars, vår ScrumMaster tipsade mig om. De pekade på olika verktyg man kan ha, vi har ju ett bra och fungerande verktyg. Man kan köra via Messenger köra remote screen och kolla någon persons kod direkt. Så att det finns inga direkta hinder för kommunikationen. Men allting tar lite längre tid och det är just det här med tids fönstret där man kan kommunicera direkt är inte stort. Speciellt nu vid vintertid, vi vinner en timme till våren!

**Intervjuare:**

**Tack så jättemycket, så mailar jag över det till dig för att läsa igenom sen.**

DJ:

Tack!

## **Bilaga 14, Martin Sternebring, medlem i Scrum Team**

**Intervjuare:**

**Intervju med ScrumTeam-medlem Martin Sternebring den 24:a januari på SonyEricsson. Du kan väl börja lite kort om dig, namn, hur länge har du varit i organisationen**

MS:

Martin, jag har jobbat på Cybercom i Malmö och varit på SE sedan oktober. Så jag har inte varit här så länge.

**Intervjuare:**

**När du kom hit, började du arbeta med detta projekt direkt?**

MS:

Mmm. Jag var inte med på själva uppstarten av själva Scrum utan jag kom in på andra sprinten.

**Intervjuare:**

**Hur lång erfarenhet har du av ditt nuvarande yrke? Vad har du jobbat med tidigare?**

MS:

Jag har jobbat sedan 2004 med kravspec, arkitektur, utveckling och teknisk projektledning.

**Intervjuare:**

**Vad har du för tidigare erfarenhet av systemutvecklingsmetoder?**

MS:

Det är den vanliga, projektmodellen eller vad du kallar det.

**Intervjuare:**

**Där man sätter upp en plan och sen kör man på den?**

MS:

Ja men innan Cybercom jobbade jag på ett företag med täta kundkontakter, det var ju väldigt agilt. Det var någon sorts projektmodell då som jag pratade om, men varje vecka kunde i princip kravspecen ändras så det blev ju agilt och det var väl ibland lite scrummish-aktigt men med projektledare som var ScrumMaster, så det blev ju inte optimalt. Han var ju både projektägare och ScrumMaster och projektledare. Så det blev ”det här är viktigt det ska vi hinna med så gör det”, han gjorde ju tidsuppskattningar också. Det var ju agilt men ändå inte, vattenfallsmodell med iterationer.

**Intervjuare:**

**Upplever du att de tidigare erfarenheterna påverkar ditt jobb med Scrum?**

MS: Mmm, det gör det.

**Intervjuare:**

**Att man har det i bakhuvudet?**

MS:

Ja vissa saker är ju ganska lika, skillnaden är att man ska börja tänka att teamet ska göra tidsuppskattningarna istället för en projektledare gör det. Tidigare så känner jag att man tappar lite kontrollen på projektet kanske, men samtidigt är det teamet som ska göra det så har man ett team som vill jobba så borde det bli lika bra på något sätt om inte bättre då. Jag ser lite för och nackdelar med det, det gör jag.

**Intervjuare:**

**Vad har du fått för utbildning om Scrum och arbetsprocessen där, och vem har givit den?**



MS:

Scrum har jag väl fått på mitt förra jobb, där var det egen läsförmåga helt enkelt. Och här har det varit via Lars som har gått ScrumMaster-utbildning, Anders har hållit i någon Scrumintro, själva processen då. Sen är det learn by doing.

**Intervjuare:**

**Känner du att du hade behövt utbildningen tidigare, att Lars nu gick den här för någon månad sen. Känner du att du hade gått annorlunda om du fått den tidigare, hur uppfattar du det?**

MS:

Ja det är möjligt, första sprinten klarade vi knappt 40%, tror jag av det totala. Men detta var en försöksperiod på initiativ från Anders så då testade vi det här Och då är det lugnt och man bör inte gå en massa utbildningar om man inte ska ha det, och det köper jag. För även utan att ha fått utbildning kan man ju känna att en metod är bra eller så.

**Intervjuare:**

**Kan du ge exempel på problem som var i de första sprintarna?**

MS:

Jag tror, jag var inte med, jag var med i andra sprinten när jag kom in. Men problemet som jag förstod det var att produktägaren och Anders då, teammanagern då, var de som satte tiderna ganska mycket. Och det gjorde att teamet inte var med på det. Från tidigare erfarenheter vet jag att tidsuppskattningar inte är lätt, men kan säga att det tar en eller två timmar på en grej, men det tar ofta mer för att man ska komma in i det och så vidare, man ska ändra och man ska dokumentera, så jag tror att det var därför att det sprack. Men när jag kom in var vi noga med att de inte ska sätta tidsuppskattningar, och då kom vi upp i 60 procent eller något. Jag tror det är lite så att om en annan sätter tiderna tar man det inte lika mycket på allvar. På allvar gör man men man kan lätt säga att jag inte satte den tiden där, det visste jag från början.

**Intervjuare:**

**Hur tror du teamet påverkades i andra sprinten då de satte estimaten själva? Och dom inte nådde fram då heller. Tror du teamet påverkas av det?**

MS:

Jag tror det kan ha effekten för i början var det jättesnålt med tid, då vågar ingen liksom styra upp någonting. För då känns det som att det inte är min roll att göra klart det på utsatt tid. Det tror jag spelar roll. Så den förmågan att sätta en tid tror jag ökar, och förra sprinten nådde vi ju ändå fram.

**Intervjuare:**

**Tror du så att säga, att själva ScrumMaster och linjchef hade kunnat göra något för att ni skulle estimerat bättre i början, så ni hade sluppit de här problemen?**

MS:

Nej jag vet inte faktiskt, tror knappt det själv för att de som inte har jobbat är det kanske ett nytt sätt att tidsestimera. Det krävs faktiskt erfarenhet, att veta det är inte bara en grej man ska göra nu så är det klart, jag gör en ny if-sats där så är det klart, det tar 10 minuter. Jag tror ändå mer att det är en erfarenhetsgrej. Och det då kanske är en nackdel med Scrum, att det inte alltid är ett team som har kompetensen att tidsuppskatta, det krävs erfarenhet och det kanske inte teamet ska behöva ha till en viss del. Men.. Samtidigt är det ju så, om de inte kan det är det skitsvårt att planera ett projekt, så liksom så det är ändå ett problem för teamet

**Intervjuare:**

**Använder ni planning poker eller något sådant?**

MS:

Vi ska det, men inte ännu. Det ska vi.

**Intervjuare:**

**Förutom tidsestimeringen, var det andra problem framförallt i början, som var uppenbara?**

MS:

Ja de där daglig mötena tog för lång tid, att folk gick in på sina lösningar. Istället för att bara säga vad de gjort och ska göra.

**Intervjuare:**

**Blev det för mycket diskussion?**

MS:

Ja det blev det, det tog 35-40 minuter, nu är det, ja dels är det Lars som är pang pang pang, stenhård och sen har nog folk lärt sig, man säger vad man gjorde, några problem, så kan man ta det sen.

**Intervjuare:**

**En transformeringstid?**

MS:

Ja man vet vad man ska säga liksom, och man vill ju berätta så kort som möjligt, så det blir nu bättre effekt tror jag.

**Intervjuare:**

**Vilka agila tekniker använder ni i utvecklingsarbetet idag så att säga?**

MS:

Hur menar du då?

**Intervjuare:**

**TDD, Continuous Integration och så, Unit Test?**

MS:

Continuous Integration är i testfasen tror jag nu så det ska vi börja använda. Om det ger effekt så att säga.

**Intervjuare:**

**Känner du att ni behöver det?**

MS:

Jag har inte sysslat med det innan så jag vet faktiskt inte. Vad är det man vinner på det förutom att man vet status på ett bygge?

**Intervjuare:**

**Har ni haft problem med att någon har checkat in saker och sen bygger det inte?**

MS:

Jag tycker inte det.

**Intervjuare:**

**Du tycker inte det?**

MS:

Nej jag tycker inte det men jag har inte kört det så. Men jag kan tycka ibland att det får inte bli att man till överkill sitter och kör FXCOP en gång i minuten. Men jag tror att om det ger i slutändan en bättre produkt utan för mycket tid går så är det ju bra. Och Unit Test har vi börjat köra nu och införa. Vi har på alla andra projekt men inte på det stora projektet. Det var det inte från början och sitta med det och göra det nu går inte, men vi ska börja göra en del där.. Men vi går mot TDD fasen, och det är väl Unit Test och. Är Continuous Integration med i TDD?

**Intervjuare:**

**Ja det har jag för mig.**

MS:

Ok så det är väl det som vi går mot.. Vi vill hela tiden kolla är det värt den tiden det kostar.

**Intervjuare:**

**På vems initiativ kommer denna typ av saker från?**

MS:

Det är Anders vad jag vet i alla fall.

**Intervjuare:**

**Har du några exempel där ni utvecklare har kommit med, att ni säger vi vill jobba så här? Ni saknar ett verktyg eller liknande?**

MS:

Ja, det gjorde ju Jonas Granström, Granström heter han. Det gjorde han förra veckan, då tog han upp det här med NDepend, som analyser koden, som man kan köra bygg och så. Han är väl lite drivande med CI och byggservrar och det, har man något så säger man till.

**Intervjuare:**

**Har ni stöd för dessa idéer då?**

MS:

Oh ja, det har ju Anders uttryckligen sagt "Absolut", och som jag har förstått det så ligger vi ganska långt i framkanten med detta.

**Intervjuare:**

**Är det en morot, att man har den möjligheten att påverka?**

MS:

Absolut, det gör ju hela tiden att man tänker att man kan förbättra något, hade det inte funnits hade man förmodligen haft tråkigare på jobbet kan jag säga. Absolut. Anders har ju en dragning någon gång i månaden, varannan vecka vad det kan bli, om en ny teknik som han har hittat. Så får vi bestämma om vi tycker det är bra eller dåligt, så får vi testa det. Har vi hittat en teknik som vi tycker är nice så har vi en dragning, så det är ju öppet, ett öppet klimat får man säga.

**Intervjuare:**

**Vi har fått uppfattningen att han (Anders) är ganska drivande med de här grejerna, upplever du det som ett problem att han är drivande och inte ni med sådana här saker?**

MS:

Nej inte ett problem tycker jag, har vi mycket att göra så kanske inte vi hinner tänka på det. Man kanske vill och tänker på det men man har inte tiden, så det är ju bra att någon hinner göra det.

**Intervjuare:**

**Vad tror du händer om han slutar med teamet så att säga, han byter jobb?**

MS:

Ja det kan ju ha en liten avstannande effekt kan man tänka sig, beror på vem som tar över. Men det känns att han är lite mer drivande än normalt, men det kan ju bero på att han kommer från utvecklarbakgrunden och tycker det är kul och så.

**Intervjuare:**

**Angående miljö hur tycker ni har det i miljön, arbetsmiljön?**

MS:

Tycker den är bra, oftast. Man kan snacka med varandra, men personligen stör jag mig inte så mycket. Vill jag inte lyssna så gör jag inte det, det är bra att man kan prata med varandra. Man hör något i örat som man har en lösning på så kanske man springer iväg dit, det fungerar bra tycker jag. Nu ska vi flytta nästa vecka, då ska de

delar upp teamet i två olika delar, men jag tror det blev så att vi hamnar på samma yta. Lars har haft det önskemålet så ska man göra Scrum så ska man sitta ganska tätt, så man kan ha utbyte utan att gå iväg, det tror jag är ganska bra.

**Intervjuare:**

**Hur är det att ha Product Owner så nära er istället för i en annan byggnad, eller annat företag, tror du?**

MS:

Han hör lite mycket ibland, kring våra lösningar och så där. Jag vet inte nackdelen med det.

**Intervjuare:**

**Kommer han ofta till er?**

MS:

Han får inte gå till oss direkt det får han inte, då måste han ha det på ett möte, Eller gå via Lars och säga vill jag byta ut något här.

**Intervjuare:**

**Har han gått till er direkt?**

MS:

Ja innan gjorde han det mer, men nu det är ju mer om det man jobbar med just nu, om vi har frågor, och det är ju fördelen med att sitta så nära att man kan väldigt snabbt få reda på faktiskt vad det är han vill med någonting. Så jag ser det som bra.

**Intervjuare:**

**Men du tycker arbetet med PO fungerar bra?**

MS:

Ja det tycker jag.

**Intervjuare:**

**Hur upplever du det fungerar att planera en sprint i taget, ser du nackdelar med det?**

MS:

Ja det är väl helhetsperspektivet, som jag har förstått det ska väl Product Owner informera om agendan övergripande, lite mer. Vilket han inte har gjort, vilket gör att man kanske inte vet vart man är. Men samtidigt är det hans uppgift att prioritera det som ska göras i sprinten, och det gör han. Men visst hade det underlättat om han kunde berätta lite mer långsiktigt. Dels kan han också göra fel, och teamet inte göra det som är rätt utifrån den riktiga prioriteringen. Om två månader kanske man ska skriva om hela logiken och så vidare, och det kan jag sakna att inte få med.

**Intervjuare:**

**Upplever du att ni efterföljer Scrumprocessen, är ni noga med att fylla i sprint backlogen för den här sprinten med att ta ner timmarna?**

MS:

Mmm.

**Intervjuare:**

**Ni slarvar inte?**

MS:

Nej, det tror jag inte, alla ska göra det innan de går hem. Annars så ser man ganska snabbt, Anders har satt upp en skärm mitt i där man ser de där kurvorna så det är nog bra.

**Intervjuare:**

**Tycker du att ni har tillräckligt med verktyg för att stödja processen då?**

MS:

Ja det tycker jag, det behövs inte så mycket mer än det. Som jag har förstått det ska man inte behöva ett verktyg alls.

**Intervjuare:**

**Tycker du det är jobbigt med verktyg?**

MS:

Det är rätt praktiskt med verktyg, skulle alla dela samma Excel-ark så skulle det vara jobbigt, skriva över och så. Men annars, det är ju bra att det finns nu. Det finns inga verktyg jag saknar nu i alla fall, samtidigt vet jag inte om det finns några bättre.

**Intervjuare:**

**Inom gruppen sker det någon inre organisering av interna roller? På mindre nivå än teamindelningen?**

MS:

Hur menar du då?

**Intervjuare:**

**Att någon alltid tar en specifik uppgift?**

MS:

Nej det tycker jag inte det är faktiskt, jag får säga att vi tar det som ska prioriteras. Men sen är det ju att man vet att en annan har gjort något liknande precis så lämnar man den eftersom han kommer göra den tre gånger snabbare. Alternativt att gör man en uppgift så kanske man vet en som gjort liknande och då lånar man honom lite. Men några roller tycker jag nog inte. Det är det som är fördelen med dagliga möten, att man vet vad alla jobbar med. Vilket gör att när man själv kommer till någonting så kommer man förhoppningsvis ihåg vem som jobbade med det, så det tycker jag fungerar bra.

**Intervjuare:**

**Den här transparensen du pratar om, tillskriver du det Scrum, eller tillskriver du gruppen det?**

MS:

Jag vet inte, jo med Scrum kanske det påtalas mer att det ska vara en del av jobbet, utbyte och så. Men jag har alltid gjort så när jag jobbat så på mig spelar inte Scrum in där, det tror jag inte.

**Intervjuare:**

**Du nämnde tidigare att agila principer kräver mer av utvecklarna, att det krävde effektivare och bättre utvecklare. Har du upplevt det problemet här? Att vissa kanske är experter?**

MS:

Nu här går vi från ett Scrumteam till två, och då blev det så att de som inte arbetat med Scrum gick till olika team. I vårt team kom in, och två som kom i vårt team hade jättesvårt, de kunde inte uppskatta timmar alls. De hade inte jobbat så, så det är jättesvårt för dem. De hade ju ingen aning liksom, så vi fick uppskatta åt dem. Då får man ju det problemet, att de kanske inte tror på timmarna för det tror det ska ta längre tid när de börjar. Då försökte vi hjälpa dem och så, men det är jättesvårt. Det vore bra att få fram en formel för att när en ny person kommer in hur han tidsestimerar och hur man löser det. Dels tidsestimerar åt honom är svårt för man vet inte hur lång tid han tar. Dels tar det resurser från de andra, en viss procent då, och det vet vi ju inte, det kommer bli jättesvårt den här sprinten, det kommer bli jättesvårt, vi får se, som ett försök nu denna sprinten.

**Intervjuare:**

**Planning Poker kanske?**

MS:

Ja det ska vi ju testa, det är ju det att man diskuterar fram en lösning.

**Intervjuare:**

**Har ni haft utbyte av andra då, när det kommit in erfarna? Har det varit problem med tidsestimeringen då?**

MS:

Nej inte alls, så det är ju inte då Scrumrelaterat utan mer kunskapsrelaterat då. Förstår man systemet, eller har erfarenhet av tidigare system så kan man då tidsuppskatta. De ser ju inte det här, vi ska ha ett nytt [ohörbart] på en folder, det vet ju inte en ny någonting om, eller förstår, det säger ingenting. Det kan ta en dag eller fyra veckor, så det problemet kommer vi nog alltid ha tror jag.

**Intervjuare:**

**Hur ställer sig teamet till att splittras i två?**

MS:

Jag vet inte, lite tråkigt kanske, för i sista sprinten gick det riktigt bra, då var vi över lägsta nivå och alla kunde systemet, nu kommer det bli om på nytt igen, men det är nog bra ändå.

**Intervjuare:**

**Varför splittade ni teamet?**

MS:

För att det var tre som var utanför. För vi har ju tre fyra verktyg.

**Intervjuare:**

**Den där slasktratt?**

MS:

Ja precis, så skulle vi ha två team som tar från båda. För de här tre jobbade ensamma då, och det gynnar inte någon. Jag tycker det är kul att de kom in, nu med ett nytt Scrumteam och hur får vi in dem i det och så. Vi sitter i samma lokaler och så, så vi kommer ju se varandra.

**Intervjuare:**

**När ni splittrades hade ni något att säga till om vilka?**

MS:

Det hade vi kanske, jag vet inte, vi fick mail att de hade planerat det, och det var Anders och Lars som bestämde det. Som jag förstod det så var vi indelade nu så att det skulle komma lite kunskap i båda teamen. Och det tycker jag är helt rätt.

**Intervjuare:**

**Har du exempel på hur ni förändrat ert sätt att arbeta efter ett Sprint Retrospective?**

MS:

Sätt att arbeta, hmmm. Alltså nej.

**Intervjuare:**

**Mer detaljer?**

MS:

Alltså vi har lärt oss olika saker, men att tidsuppskatta, ja där har vi lärt oss vad som tar tid, i vissa delar tar det ju mer tid. Men arbetssätt vet jag inte, jag har ju bara varit här ett par månader och jag har ju arbetat ungefär så här tidigare. Det kanske någon annan kan svara bättre på.

**Intervjuare:**

**Har det uppstått några konflikter i arbetet?**

MS:

Nej, inte vad jag vet, vi kanske är konflikträdde?

**Intervjuare:**

**Inte lösningsmässigt, att ”vi ska göra så här, nej gör så här”?**

MS:

Nej, jag tror inte, inte vad jag kan komma på. Om man hjälper någon så ger man, man går ju inte in och löser problemet utan ger tips och ledtrådar hur de löser det. Går det igenom testerna så är det ok.

**Intervjuare:**

**Är det en relativt homogen grupp? Om hur arbetet ska skötas, har ni liknande värderingar?**

MS:

Ja det tycker jag nog, när vi har ett problem så gör vi en hållbar lösning, det är inte ett snabbhack.

**Intervjuare:**

**Upplever du att ni är homogena i form av vad ska man säga åsikter och sådana bitar?**

MS:

Värderingar? Ja det tror jag, de jag har hunnit prata och lära känna tycker jag nog. Alla är ganska normala.

**Intervjuare:**

**När går ni som team till ScrumMaster?**

MS:

Antingen att vi inte hinner med, då diskuterar vi vad vi ska prioritera och plocka in och så, eller om det är något förhinder. Jag kan inte lösa denna uppgift på grund av något annat, då är det hans uppgift att se till att man kan lösa den.

**Intervjuare:**

**Finner du det frustrerande att arbeta efter Scrum och agiltet, agila principer?**

MS:

Nej, för det är beslut som man tar, men det kan ju kännas lite överkill ibland att man absolut inte får göra annat än de som är i sprinten. För att skjuta det till nästa sprint kan kosta 10 gånger mer tid, och det hade jag som projektledare kanske inte köpt. Men man får ju acceptera det, att det är så man jobbar, om resultaten i längden ger bättre resultat. Så får man justera det, vill vi verkligen göra så? Det är som två världar som går ihop så man får försöka hitta det bästa sättet. Men jag tycker det fungerar bra med Scrum.

**Intervjuare:**

**Vill du ändra något i processen?**

MS:

Jag har nog jobbat för lite med det så här.

**Intervjuare:**

**Du känner dig motiverad för Scrum?**

MS:

Ja absolut, det är en liten tankeändring och det är det jag håller på och bearbetar. Det är för att jag kommer från mer vanlig projektbit, till det här. Och man tycker att projektägaren har lite att säga till om men det har han inte, det är ju bara det att man delar upp det, sen blir det klart. Men ja, jag tycker det är bra.

**Intervjuare:**

**Schwaber pratar kanske om det, att arvet från vanlig projekthantering är det största problemet när man introducerar Scrum.**

MS:

Ja men man vet ju inte hur allt funkar i Scrum, är det verkligen det bästa och så där. Det finns frågetecken och så. Men det finns inget jag känner är jättedåligt, inte än i alla fall.

**Intervjuare:**

**I början nämnde du att utbildningen är relativt informell, möte och att läsa själv. Känner du behov av mer formell utbildning?**

MS:

Alltså nu kör vi och lär oss genom jobbet, så det är utbildningen att företaget betalar i form av att man kanske inte är så effektiv direkt, så det är ju så.

**Intervjuare:**

**Men du har inte känt dig frustrerad eller så att du inte har koll på allt inom Scrum och så där?**

MS:

Nej det tycker jag inte, inte än i alla fall. Men det är svårt att säga, hade jag gått en utbildning hade jag känt annorlunda kanske. Det kanske man hade kunnat svara på i efterhand men.

**Intervjuare:**

**När det gäller slutet av en sprint, blir det någonsin så att ni nästan hinner klart?**

MS:

Ja, men det problemet går ju inte att komma runt på något sätt. Förra sprinten så var det kanske vid 11 och vi skulle frysa vid 12, och vid 11 var Code Review klar och fick det här ska ändras och det fanns det ju inte tid till då, för det var så stora ändringar och jag ville inte göra dem för jag visste inte hur det skulle påverka annat. Då kände jag liksom att det är risk att någonting smäller någon byter namn på någon klass och så, så jag tror vi gjorde de ändringarna ändå, och vi hade tur när vi gjorde dem. Men jag vet inte hur man ska komma runt det, sista timmarna. Alla kan ju inte sitta och rulla tummarna, alltså vi har systemtest efter så det är den stora delen och sanity check. Men det kan ju bli problem sista timmarna, om Code Review verkligen ger något som vi bör ändra. Men då får man ju ta beslut att vi får skjuta på något, men jag vet inte hur man ska komma runt det. Det är som det är.

**Intervjuare:**

**Bästa av bästa världar, om ni hade haft 95 % Code Coverage. Med Unit Test, och CI hade ni checkat in då och gjort de ändringarna tror du?**

MS:

Ja det hade jag gjort.

**Intervjuare:**

**Så verktygen hade hjälpt er med stress på slutet då?**

MS:

Ja det hade det ju.

**Intervjuare:**

**Mmm.**

MS:

Ja då får man ju en grundtrygget i det så att ja..

**Intervjuare:**

**Ni har inte råkat ut för att ni närmar er slutet av en sprint och bara har stora grejer kvar?**

MS:

Jo det hände första sprinten.

**Intervjuare:**

**Så ni kom fram till att det finns ingen del jag kan göra för jag hinner inte klart den?**



MS:

Ja, eller att det är en stor grej, som tar 30 timmar. Men det är bara en person som kan göra den, då går det inte ihop. Då har vi infört att på vårt möte när vi planerar sätter vi preliminära personer på de stora uppgifterna. Så jag börjar kanske med den här som är på 15 timmar och du tar den som är på 20. Mmmm.. Vet inte om det är Scrum men det gör att man verkligen börjar med dem och hinner med, och det är ganska bra att ta de svåra först, eller det stora jobbet först. För att får du problem där kan du tidigt säga att du har problem med tiden så kan du prioritera om. Vi försöker inleda med de stora grejerna.

**Intervjuare:**

**Ni har indikerat på att ni inte vet om ni bryter mot Scrum eller inte?**

MS:

Det kan man inte göra, det finns väl inga fel i Scrum?

**Intervjuare:**

**Tror du att ni har någon oro i detta, att utvecklarna undrar fan är det här Scrum. En press att ni ska anamma det fullt ut?**

MS:

Pressen finns ju där, i och med att vi har uttalat att vi ska köra Scrum. Men jag tänker inte i daglig verksamhet om jag kör Scrum eller inte. Jag tror inte de gör det. Utan jag följer väl liksom huvudmallen sen, ja.

**Intervjuare:**

**Har du någonsin planerat mer än en sprint i taget i huvudet?**

MS:

Nja..

**Intervjuare:**

**Du har kommit in i tänket?**

MS:

Nej det har jag nog inte, nej. Utan man kör i nuet, och det känns bra. Det jag saknar är när man kör många projekt så är det väl överblicken som saknas, lätt att tappa det stora hela, det tror jag nog är det som kanske kan skilja sig åt. Jag har inte tänkt så mycket på det. Det är en stor del skillnad, det finns i Scrum också att man inte planerar mer, tidigare var det så, att om tre månader ska vi vara där. Men på vägen var det mycket som ändrades och byttes ut. Och det är ju att på vägen nu kallas sprint.

**Intervjuare:**

**Ja tack, vi har tagit upp det mesta, tack för du tog dig tid.**

MS:

Tack så mycket själv.

## **Bilaga 15, Karl Eklund, medlem i Scrum Team**

**Intervjuare:**

**Intervju Scrum på Sony Ericsson 2008-01-31. Om det är ok, kan vi ju börja lite kort med namn och tid inom företaget, position osv.**

KE:

Japp. Mitt namn är Karl Eklund och jag började som underkonsult på Sony Ericsson april 2005 och ingick då i en extern testgrupp, ett år senare flyttade jag in på Sony Ericsson för att utveckla automatiserade testskript för det interna verktyget Batch Remote Automated Testing, och kom då i kontakt med Anders Nybergs grupp. Jag jobbade där som skriptutvecklare fram till i somras, då vi efter en omorganisation kom ut här, och har mer och mer integrerats till att bli utvecklare inom verktyget, och i höstas blev jag verktygsutvecklare på heltid. Jag sysslar huvudsakligen med processer och use-case och test-case. Men ska nu lära mig att faktiskt utveckla.

**Intervjuare:**

**Det låter bra. Vad har du för tidigare erfarenheter av systemutvecklingsmetoder?**

KE:

Inga alls.

**Intervjuare:**

**Har du jobbat på liknande sätt innan?**

KE:

Nej. Jag har ju utbildning i det från 2001 till 2005, men inget arbete inom den branschen.

**Intervjuare:**

**Så all tidigare erfarenhet av systemutvecklingsmetoder är enbart teoretisk?**

KE:

Japp.

**Intervjuare:**

**Upplever du att de teoretiska erfarenheterna hjälper dig någonting i ditt nuvarande jobb med Scrum?**

KE:

Ja och nej. Teori har en obehaglig tendens att inte gå att översätta direkt till praktik. Det är alltid hänsynstagande man behöver göra i praktiken som man inte behöver göra i teorin. Teorin är kall och klinisk och det är ingen som står och säger att det här kostar tio miljoner och jag behöver ett resultat imorgon. Och det är det stora problemet man har, men det är också den stora fördelen, för att man kan ta ett steg tillbaka och säga att metodiken eller processen är utvecklad såhär, och det finns en anledning till det och när vi avviker från den så minskar vi värde eller minskar effekt, och kan man då bygga ett business-case och förklara att i slutändan kommer detta kosta 100 miljoner i underhåll, det räcker med hundra tusen i underhåll, som är onödiga och som vi kan spara in nu innan vi går för långt. Så teorin hjälper väldigt mycket men den stjälper också om man har lite rosenfärgade glasögon på sig, man vill att allting ska funka på ett visst sätt och det gör det inte.

**Intervjuare:**

**Vad har du för utbildning om Scrum?**

KE:

Ingen.

**Intervjuare:**

**Men här på företaget?**

KE:

Vi har haft orienteringar, korta entimmarsmöten, då vi i princip har gått igenom iden bakom Scrum, jag tror det var tre stycken sådana möten.

**Intervjuare:**

**Har det gällt abstrakta saker eller mer konkreta detaljsaker i Scrum?**

KE:

Nästan bara abstrakta. Vi har valt att se Scrum som en metodik snarare än en metod, så om man då siktar på målen, t.ex. att ett av målen är att isolera utvecklarna och sådär, då känns det bättre att hålla den teoretiska delen för de flesta, och för de som behöver ta det praktiska arbetet så får de lära sig de detaljerna.

**Intervjuare:**

**Var du med från början med Scrum?**

KE:

Japp.

**Intervjuare:**

**Så du var med i det stora åttamannateamet?**

KE:

Nej, jag var inte med från början, jag var med på avdelningen men just då hade jag en för stor arbetsbörda som kodgranskningsansvarig för skripten, så jag hade helt enkelt för mycket fasta uppgifter.

**Intervjuare:**

**Men du har insikt i hur arbetet med Scrum förändrades från början, med tiden så att säga?**

KE:

Absolut. Den stora skillnaden är ju att den första ScrumMastern var ju sektionschefen Anders, och det var ju ett jätteproblem, bland annat för att det ju var mot honom som ScrumMaster skulle skydda oss. Och hur duktig du än är på att ha två hattar så är du fortfarande bara en människa, och hans prioriteringar som sektionschef slog ju igenom hans ledning som ScrumMaster. Vi hade också väldigt dålig kunskap, kändes det som, i skillnaden mellan Backlog och Scrum-items. Så det var väldigt svårt, tyckte jag, det första Scrumplaneringsmötet jag var med om, att förstå meningen med att sätta timmar på allt var eftersom det inte blev en fast lista ändå. Och det har ju löst sig. Så det är väl den stora förändringen, att chefen inte är inne i Scrum längre.

**Intervjuare:**

**Men du anser att organisationen stödjer ert arbete med Scrum?**

KE:

Japp.

**Intervjuare:**

**Men inte bara linjeföraren, utan även övriga organisationen?**

KE:

Övriga har ingen insyn i vårt arbete.

**Intervjuare:**

**Så de kommer inte och påverkar?**

KE:

Nej. Skulle de cheferna ett steg upp lägga sig i, de har ju hundratals människor under sig, de skulle aldrig kunna göra micro management på det sättet. Så att de lyckas hålla fingrarna ur syltburken är ju snarare status quo, om han hade börjat med micro management, då hade vi haft ett riktigt problem.

**Intervjuare:**

**Hur upplever du att samarbetet med ScrumMaster fungerar?**

KE:

Vi provade ett tag att ha en roterande ScrumMaster, att i varje Sprint ha en ny ScrumMaster, och det funkar inte. Detta för att några av de personerna som skulle vara ScrumMaster är för timida för att ha den rollen, man måste vara redo att stå upp och säga antingen ”nej, jag kan inte ta det här beslutet för teamet, och du får inte prata med dem, vad har du med det att göra? Du är chef, du har gett oss de här jobben, håll dig undan.”. Det funkade inte, och jag kommer mycket väl överens med Lars, han är hård och han har lätt att brusa upp och så, men han är alltid ärlig och han är alltid öppen, och det funkar. Men man får inte vara för feg i närheten av Lars, för han har som sagt en tendens att brusa upp och är man rädd för det, då kan det vara lite svårt. Men jag tycker i det hela att det funkar väldigt bra, och inom gruppen är vi ganska måna om att fånga upp det där, när Lars är på dåligt humör så är det inte de klenaste som går fram och pratar med honom. Men den gruppdynamiken måste man ju ha oavsett var man är.

**Intervjuare:**

**Men han är inte för kontrollerande eller så?**

KE:

Absolut inte. Han är snabb på att identifiera när folk försöker.. maska är fel ord, men när folk inte klarar av att lösa uppgiften, när det börjar dra ut på tiden, när folk är osäkra, så är han snabb på att påpeka det, och peka ut någon: ”du gör inget just nu, går över och prata med honom”, i de fallen där folk inte själva säger att de behöver någon. Så vi behöver det nu i början, vi behöver en liten spark, eftersom det är ett ovant sätt att arbeta. Att gå från universitetet är nog enklare, för grupparbetena på universitetet är väldigt Scrumliknande. De flesta föreläsare skiter fullständigt i vem det är som har skrivit en viss rad kod eller dokumentation, bara den finns där. Så är det normalt inte i utvecklingsarbete på en arbetsplats, där säger folk ”jag hade det här jobbet, jag gjorde det jobbet”. Men nu är gruppen ansvarig för den kompletta leveransen.

**Intervjuare:**

**Hur tycker du att samarbetet med Product Owner fungerar?**

KE:

Vår Product Owner har nog inte riktigt vant sig vid att inte kunna gå in och detaljstyra under utvecklingsarbetet. Jag tror att vi är på god väg till det, men ibland känns det som att han inte förstår att det är andra saker han behöver göra, han behöver tänka två veckor, fyra veckor, sex veckor i framtiden, hela tiden. Han måste uppdatera Product Backlog, och inte bara ta in nya items, utan ”jaha, nu har situationen förändrats, den här sprint-item löstes på ett visst sätt, då måste jag uppdatera de här Backlog items”. Han har sagt det flera gånger, att han inte kan hålla allting i huvudet, och det är sant, det kan han inte, därför måste han utveckla någon form av rutin, att gå igenom det han har. Och han är inte riktigt där ännu, därför blir det lite osynkat mellan honom och vår ScrumMaster.

**Intervjuare:**

**Men det är något som har blivit bättre sen start?**

KE:

Vi hade ju inte riktigt det problemet, eftersom för inte alls länge sen hade Product Owner väldigt stort inflytande även under Sprinten. Det har blivit lite bättre, men jag tror att om två Sprintar, så kommer det att vara mycket bättre, eftersom Lars är så bra, och är så bra på att säga ”jag tycker inte att detta är en bra lösning”.

**Intervjuare:**

**Så det krävs en stark ScrumMaster för att styra allt på rätt sätt?**

KE:

Det krävs en stark ScrumMaster tror jag, för att starta upp det i en ovan miljö. Jag tror att så som vi började med att ha någon som läste lite om det och så, det kan vara en god ide att testa för att se ifall processen fungerar överhuvudtaget, men sen gäller det att ha någon med lite skinn på nästan, och teknisk färdighet, det ska man inte underskatta. Man får inte ha en dum ScrumMaster, han måste ha tillräckligt god förmåga att kunna titta på utvecklarna han jobbar med och kunna säga ”nu snackar du skit, nu är det någonting du maskar med, något du döljer”. Han måste ju vara lite hönsamma, och då måste han ju ha färdigheterna att titta på sina små kycklingar och veta vad det är som pågår. Han behöver inte vara den bästa utvecklaren, men han måste förstå språket.

**Intervjuare:**

## **Hur tycker du det fungerar att planera en Sprint i taget?**

KE:

Jag önskar att vi skulle ha lite mer framförhållning. Mest för att jag har ett kontaktansvar mot dem som använder verktyget, och när de ställer mig frågor så vill jag gärna veta inte bara vad vi jobbar med den här två veckorna, utan hur ser det ut om fyra veckor. Och som det är nu, som jag påpekade förut, med vår Product Owner, så får vi inte riktigt det stödet ur backlogen. Det är lite för osäkert.

### **Intervjuare:**

**Nu ska vi se här. Jämförelser med tidigare yrkeserfarenheter kanske vi ska skippa.**

KE:

Nja, alltså jag började ju som testare, och sen blev jag skriptutvecklare och nu är jag systemutvecklare, så jag har ju haft tre olika yrkesroller under de åren jag jobbat här, och jag kan ju säga att det monumental skillnad att vara systemutvecklare och skriptutvecklare, det finns ingen jämförelse. Då gör du allting "on the fly", allting hänger i luften hela tiden, systemutveckling är ju mycket en mycket mer kontrollerad verkstad.

### **Intervjuare:**

**Mer strukturerad?**

KE:

Absolut! Saker som man kan komma undan med en "try catch"-variant i skript kan du inte komma undan med i systemutvecklingen eftersom där finns verktyg som undrar "vad gör du här för något?", och det är så jag vill jobba, så jag är mycket nöjd allt som allt.

### **Intervjuare:**

**Vad anser du är för- och nackdelarna med Scrum?**

KE:

Den stora fördelen är att folk tar ansvar för det som gruppen gör. Det är nog den absolut största fördelen jag upplevt med Scrum, den andra fördelen är att du får upp ett tempo utan att egentligen... Alltså du har ju en tydlig lista med arbete, och om du fastnar med någonting så vet du att du kan vända dig till gruppen och säga "jag har fastnat med det här, vi jobbar alla mot samma mål, någon måste hjälpa mig med det här". Det dåliga är, att om du är osäker på det du gör, och behöver lite hjälp, så känns det väldigt mycket, och det är i den situationen jag är i nu, jag är inte duktig på C#, jag kan inte C# och .NET, och då måste jag fråga så väldigt mycket om hjälp, och varje gång jag frågar om hjälp så tänker jag "nu dubblad jag tidsdebiteringarna för det jag jobbar med", så i princip vad jag än börjar med som har med utveckling att göra, så blåser jag tidsestimaten direkt, för jag måste ha någon bredvid mig som drar i trådarna, så det är väldigt lätt att känna sig skyldig. Å andra sidan så är det ju meningen med Sprintplaneringen att man ska kunna bygga in tillräckligt mycket löstid fram och tillbaka på detta. Det är lite svårt att göra, det finns säkert metodiker eller tekniker för det, men det är lite svårt. Den andra stora nackdelen jag ser är att det känns väldigt svårt att bryta en Sprint, eftersom du inte avslutar en item i taget, utan flera items som pågår samtidigt, och vi kan inte bara säga "nu måste vi jobba med brandsläckning, vi har det här felet som vi måste lösa", för då måste vi gå in i content management och går tillbaka till en gammal frys, göra en ny bransch, utveckla på den tills felet är löst, merga tillbaka den till den utvecklingsbas man höll på med förut, släppa den som release: "nu har vi löst det problemet och här har vi den", och sen startar man om en ny Sprint med halvfärdiga items, och det känns lite klumpigt, eftersom uppstartningsarbetet på en Sprint är ganska tungt.

### **Intervjuare:**

**Det känns onödigt att repetera det?**

KE:

Japp, och det finns inget bra sätt att göra "Jaha, nu var vi här, nu kör vi vidare, för så funkar inte den mänskliga hjärnan liksom, vi kan inte bara frysa en tråd, göra något annat och sen fortsätta, det funkar inte så.

### **Intervjuare:**

**Tycker du att det är svårt att tidsestimera?**

KE:

Jag tycker det är fruktansvärt svårt att tidsestimera.

**Intervjuare:**

**Är det något man blir bättre på med erfarenhet?**

KE:

Man blir ju ärligare. Vi underskattade en väldigt massa saker eftersom man tycker det känns pinsamt att säga att det två timmar att skriva ett test-case. Det är ju bara att kladda ner lite på ett papper, du har ju skrivit test förut. Visserligen, men man vet inte när man börjar skriva ett test-case, det är svårt att veta detaljnivån på testet, hur många smådetaljer behövs det egentligen för att klicka på en knapp? Det behövs ingenting, men om du ska hitta knappen också, och om du måste skaffa ett state i verktyget eller datorn du jobbar med, det kan ta väldigt mycket tid att skriva ett bra test-case för att du alltid ska kunna återskapa ett visst state. Det ska ju vara generellt så att små ändringar i verktyget inte negerar ditt fall. Man måste bli väldigt ärlig i tidsestimeringen, och det är den statusen vi är i nu. Sen måste vi dessutom bli bättre på själva tidsestimeringen av nyutveckling och problemkomplexitet, och det är nästa steg. Så jag tror det går väldigt mycket i plåtår i utvecklingen i färdigheter. Jag tror det är svårt att både bli ärlig och få in färdigheterna samtidigt, det är väldigt svårt.

**Intervjuare:**

**Du nämnde innan att be om hjälp och så, har ni bra och öppen kommunikation tycker du?**

KE:

Ja, det tycker jag.

**Intervjuare:**

**Är det bra att arbeta i en så öppen miljö?**

KE:

Jag har alltid gjort det. Jag har aldrig suttit på kontor, och jag tror inte jag hade passat på kontor, alldeles för stor risk att ifall jag fastnade på någonting skulle jag antingen bara sitta och stirra på texten, eller så skulle jag surfa på någon nyhetshemsida. Jag tror att en öppen miljö är rätt så bra.

**Intervjuare:**

**Det finns ingen risk att man blir distraherad när andra pratar?**

KE:

Jo. Det är därför man har hörlurar. Om man gillar klassiskt eller hårdrock eller vad som, bara man har en liten konstant ljudnivå i hörlurarna, så reagerar du inte på tal på samma sätt. Samtidigt är det ju också ganska skönt att kunna höra vad folk säger runt omkring dig, för du har ju informella kommunikationsvägar också, som kan vara svåra att upprätthålla ifall alla människor har sin egen enhet. Som det är nu, och någon kommer och pratar känslomässigt om något, så kan man hålla kvar händerna på tangentbordet, lyfta ögonen och spetsa öronen, så är du med i kommunikationen direkt, det är ingen som behöver sprida det vidare, och det är ingen som behöver ha ansvar för kommunikationen. Det är att välja, antingen är du med i jobbet eller i kommunikationen, direkt. Det är en stor fördel, men kan också vara en stor nackdel, det är väldigt svårt att undvika det ifall man är känslig.

**Intervjuare:**

**Upplever du teamet som en väldigt homogen grupp? Har ni liknande värderingar och åsikter?**

KE:

Nej.

**Intervjuare:**

**Det är alltså en spridd grupp?**

KE:

Ja.

**Intervjuare:**

**Men åsikter om hur arbetet ska skötas?**

KE:

Den är ganska homogen ja.

**Intervjuare:**

**Och har det någon effekt på arbetet i Scrum? Några speciella problem som annars hade uppkommit?**

KE:

Oh ja. Det finns en typ av utvecklare som jag brukar kalla "hack and slash". Det handlar då bara om att få fram resultatet. Den typen av utvecklare är oftast otroligt skickliga, och har en fantastisk kunskapsnivå och kan få datorer att hoppa upp och skälla. Men de är oftast inte bra i det långa loppet, eftersom de lämnar luckor, de är osystematiska, deras kod är svår att läsa, de använder oftast det de är intresserade av för tillfället, de använder inga homogena utvecklingsmetodiker. Den gruppen vi har nu är mycket intresserad av att det ska se bra ut, man ska ha en snygg kod, och det är något man piskats på under universitetet, som jag tyckte var så löjligt, och nu när jag hållit på en del inser jag hur otroligt viktigt det är att leverera ett arbete som vem som helst kan arbeta med.. Det är nästan viktigare än funktionalitet.

**Intervjuare:**

**Men du ser det som en förutsättning för att Scrum ska fungera?**

KE:

Japp. För om du bara slänger dig in med köttynan och hugger dig fram tills du hittar felet, så kommer det ta fem gånger så lång tid nästa gång.

**Intervjuare:**

**Ni kör gemensamt ägande av kod här?**

KE:

Japp.

**Intervjuare:**

**Och då är det ännu viktigare kan jag tänka mig.**

KE:

Ja, man ska ju inte sabotera för någon annan. Men nu är det ju så att så fort man är mer än en utvecklare så har man ju gemensamt ägande av kod i slutändan. Det är ju bara att man skjuter upp felet till en stackars cm-ansvarig istället, och hur skoj är det?

**Intervjuare:**

**När vänder ni er som team till ScrumMaster? Istället för som individ?**

KE:

Nu är det ju som så att ScrumMaster är en del av vårt team, så det är lite svårt. Men i princip så är det så att om man är utvecklare och man stöter på någonting som antingen är fel eller som man inte förstår, ett Backlog item eller något sånt där, så tar man kontakt med en annan utvecklare och frågar honom: "vad tycker du om det här?", om den utvecklaren då också säger att han inte heller förstår samlar man ihop fler utvecklare och frågar dem vad de tycker. Man eskalerar alltså i steg, och det kan vara diskreta steg, det kanske är så att man går fram till en grupp av tre utvecklare och frågar om de kan lägga en minut på det här. Så det är ju inte så att det måste vara en, två, tre och sen fyra. Och rätt så fort så inser man att alla säger samma sak, och då tar man det till ScrumMaster, och om man gör det när man är två eller om man gör det när man fyra, det spelar liksom ingen roll. När det kommer konsensus att det är någonting som saknas eller något som är ett problem, då tar man det vidare.

**Intervjuare:**

**Är det någonting du tycker är frustrerande med Scrum?**

KE:

Ja, att på morgonmötet stå och förklara varför man inte lyckats, det tycker jag är frustrerande, men det är ju eftersom jag är den svaga länken, så är det jag som står där och ber om ursäkt.

**Intervjuare:**

**Fast finns det inte ett förstående för det?**

KE:

Jo visst finns det det, men det spelar väl ingen roll? Det är ju inte kul att stå där när man är fullständigt medveten om att man kommer att stå där varje dag och säga ”jahapp, lyckades inte idag heller, jag behöver mer hjälp”, det är ju ganska svårt. Jag accepterar det, men det är ju frustrerande. Och det finns ju väldigt få situationer inom mjukvaruindustrin där du är så synlig, som på ett Scrum-morgonmöte. Annars har man ju en deadline som ligger veckor i framtiden, och inte varje morgon. Hade du en dålig dag? Syns direkt på morgonmötet dagen efter.

**Intervjuare:**

**Det är lättare att mörka sin progress annars?**

KE:

Absolut. Och det är ju mycket svårare att ha en dålig dag.

**Intervjuare:**

**Du förklarade innan för- och nackdelarna med Scrum. Vilka är enligt dig för- och nackdelarna med att jobba inom ett Scrumteam? Eller det är kanske samma?**

KE:

Ja, det var så jag svarade iallafall. Men den stora fördelen med Scrum för Product Owner när man väl kommer upp till den professionalitetsnivån är ju förutsägbarheten, att ”jag har en puck här, den är bedömd med den här komplexiteten, den är ungefär såhär många arbetsdagar, och så småningom kommer man ju att lära sig att om jag lämnar den här till teamet så kommer den att bli färdig på en sprint, för de här killarna, eller det här teamet, kan bryta ner det och jobba parallellt, istället för sekventiellt. Det leder ju då till att han kan säga ”om åtta veckor har vi den här funktionaliteten i verktyget, eller så är den funktionaliteten struken, antingen eller”. Och det är ju mycket skönt, istället för att sitta med tidsestimat, och sitta och säga att vi försöker få det gjort till t.ex. vecka åtta. Det tror jag är mycket skönt för Product Owner, när man kommer upp till den professionalitetsnivån att man kan göra bra komplexitetsbedömningar och bra tidsestimat.

**Intervjuare:**

**Ja, då är det nog inte så många fler frågor, du ska ha stort tack för din hjälp.**

KE:

Inga problem.



## **Bilaga 16, Jonas Nyberg, medlem i Scrum Team**

**Intervjuare:**

**Intervju** angående Scrum på Sony Ericsson den 24:e januari klockan 15, och då börjar vi med lite grundläggande information om dig, namn, tid i organisationen, och lite så.

JN:

Jonas Nyberg heter jag, och jag har jobbat här som utvecklare och konsult sedan juni i fjol, vilket blir sju månader eller så.

**Intervjuare:**

**Och du har varit i den här avdelningen med det här systemet hela tiden?**

JN:

Japp, med BRAT.

**Intervjuare:**

**Hur lång erfarenhet har du i ditt nuvarande yrke?**

JN:

10 år ungefär.

**Intervjuare:**

**Vad har du för tidigare erfarenheter av systemutvecklingsmetoder?**

JN:

Förutom Scrum då som är den senaste, så har det mest varit metoder som liknar vattenfallmetoden kan man säga. Ja, något projekt har ju varit inne lite på att använda användningsfall en del, men det är nog mer vattenfallsmetoden faktiskt. Vi arbetade länge med att ta fram kravspecar tillsammans med beställaren innan vi började utvecklingsarbetet.

**Intervjuare:**

**Vad har du för tidigare erfarenheter av Agila tekniker?**

JN:

Det här är den första erfarenheten jag har av det, den tiden som jag jobbat med det här.

**Intervjuare:**

**Upplever du att dina tidigare erfarenheter med systemutvecklingsmetodik påverkar ditt arbete idag på något sätt?**

JN:

Jo, det gör det ju, man har ju med sig sin historik, och med åren har man ju arbetat fram ett visst sätt att arbeta, så det har man ju såklart med sig nu även fast man försöker tänka i de här banorna med Scrum, så visst påverkar det.

**Intervjuare:**

**Kan du ge något exempel på hur det har påverkat? Någon gång du tänkt kors i stäv med hur du borde?**

JN:

Det kan vara att tänka ”men.. de här kraven är ju inte riktigt utredda redan, det här borde ju vara bestämt”, och vara skrivet i eldskrift och hugget i sten. Här är det mer iterativt och man bollar fram och tillbaka.

**Intervjuare:**

**Men det är inga jätteproblem?**

JN:

Nej, absolut inte, det är ju inte som att det är ett problem att jobba med den här tekniken på grund av det, men det är en liten inväpningsprocess.

**Intervjuare:**

**Vad har du fått för utbildning angående Scrum och dess processer?**

JN:

Ja, vi har haft lite genomgångar av Anders och Lars, där de berättat hur allting ska fungera, och att det är såhär han vill att det ska fungera för oss, sen har ju Lars gått en utbildning för ett par veckor sen, så han har ju kommit med feedback efter det, och berättat om de viktigaste punkterna därifrån.

**Intervjuare:**

**Känner du att det hade påverkat dig annorlunda om du fått informationen från början, istället för att få lite i början och sen fått höra "nu ska vi göra såhär istället". Har det varit några problem?**

JN:

Nej, problem skulle jag inte säga, men om vi jämför den senaste sprinten vi hade med den första sprinten, så har det blivit mycket bättre såklart, Det beror lite på Scrumtänket, och att vi har varit mycket mer inne i det den sista sprinten, men det är inte bara utbildningen utan det är mer erfarenhet och att man lär känna varandra och kommer in i arbetssättet mer och mer, vem som ska fixa vad och så.

**Intervjuare:**

**Känner du att du hade behövt mer utbildning i hur Scrum fungerar? Mer formell utbildning?**

JN:

Jag tror inte att det är ett måste, det är klart det inte skadar att gå någon lite längre utbildning och lära sig lite mer hur det är tänkt att det ska fungera, men å andra sidan, det beror ju så mycket på vad det är man ska lösa och för system man ska utveckla. Jag tror nästan att det väger in mer att man bollar inom teamet, vad som fungerar bra för oss, "vi löser det på detta viset". Så jag tror ändå att det är viktigare än att man har tung formell utbildning i Scrum. Jag kanske hade sagt något annat om jag gått en bra utbildning, men som jag känner nu ser jag det inte som ett måste att ha gått en utbildning.

**Intervjuare:**

**Så det är viktigare att teamet får organisera sig och lära känna varandra?**

JN:

Ja, jag tror det faktiskt.

**Intervjuare:**

**Vilka agila tekniker använder ni i dagsläget? T.ex. unit testing och continuous integration?**

JN:

Vi använder Unit Testing lite grann i BRAT-projektet, det har använts mer till andra projekt i vår grupp som jag inte jobbat i. Vi har börjat införa det, men inte så mycket än. Det är ju så att vi byggde det mesta i systemet innan vi kom in på det här med Scrum och Unit Testing, och det är ju en liten tröskel att införa i ett system som redan är byggt, jämfört med att komma med det från början.

**Intervjuare:**

**Har ni fått utbildning i det?**

JN:

Ja, vi har haft en genomgång i det av Anders. Sen har vi Continuous Integration som är på gång, Dick har tittat rätt mycket på det men det är inget som jag har sysslat med hittills, men det är ju något slags mål som vi strävar mot.

**Intervjuare:**

**Har ni haft problem kopplade till att ni inte använt Continuous Integration, att man till exempel checkar in något och går hem för dagen, och sen funkar inte bygget för nästa person?**

JN:

Jo, det har hänt, det har det gjorts. Men det brukar lösa sig rätt fort, men visst händer det att det hamnar i osynk på något sätt.

**Intervjuare:**

**Men ni själva har ansett att ni behöver det här, Continuous Integration?**

JN:

Nja, det är nog mest att Anders har varit drivande där, men det är säkert flera som har varit inblandade, jag har inte varit så inblandad där i beslutet att gå mot det, det är andra utvecklare som varit aktiva i det och mer delaktiga. Men jag tror att det skulle lösa en del problem, inte för att det är kaos nu på något sätt.

**Intervjuare:**

**När ni först införde Scrum, vilka var de största svårigheterna i den allra första perioden?**

JN:

Det var mycket frågetecken kring hur gruppen skulle fungera, och vilka regler som gäller kring en task. Alla var ju lite nybörjare på den här modellen, så det fanns olika frågetecken över saker som "när ska jag släppa den här tasken" när man kände att man inte kom vidare och de andra i teamet inte heller visste hur vi skulle hantera den. Men det är också sånt som vi blivit bättre på varje sprint.

**Intervjuare:**

**Hade ni någon workshop eller var det sånt som bara togs upp titt som tätt att ni ska vi ta tag i det här problemet?**

JN:

Det är som vi tagit upp efterhand, och även diskuterat på våra möten, vilken riktning vi ska ta och hur vi ska tänka i olika situationer.

**Intervjuare:**

**Känner ni att organisationen stödjer ert sätt att arbeta här och att ni har valt Scrum? Från linjeföraren och så?**

JN:

Ja från vår linjeförare absolut, han är i högsta grad positiv till att vi arbetar såhär.

**Intervjuare:**

**Men utanför så att säga?**

JN:

Jag tror, jag har inte så jättemycket kontakt med de utanför, men min känsla av vad det lilla jag hört, så tror jag de tycker det är bra, med den här relativt täta leveransen och att man verkligen bestämmer sig inför varje tvåveckorsperiod att de här sakerna ska prioriteras.

**Intervjuare:**

**Hur upplever du att Scrum fungerar när ni sitter så här öppet? Känner du att det skulle ske bättre om ni hade det på något annat sätt?**

JN:

Jag tycker att det fungerat bra, det finns både för- och nackdelar, fördelen är ju såklart att det går snabbt att få tag på en person när det gäller någonting, man ser snabbt och enkelt om de är på plats eller inte, istället för att sitta utspridda i olika rum. Sen kan det ju vara en nackdel om det är flera diskussioner i samma rum samtidigt och man verkligen behöver koncentrera sig på sitt eget, det är ju viss distraktion också.

**Intervjuare:**  
**Men det är inget jätteproblem?**

JN:  
Nej jag tror att fördelarna uppväger nackdelarna faktiskt, totalt sett.

**Intervjuare:**  
**Kan det vara så att man snappar upp konversationer när andra har problem?**

JN:  
Ja, det är ju en av fördelarna, det kan ju vara så att två stycken pratar längre bort om något som man råkar veta, och kan informera dem att ”jo, det finns en anledning till det där”, så det är en av fördelarna.

**Intervjuare:**  
**Hur upplever du att samarbetet gentemot ScrumMaster fungerar? Kontrollerar han, beordrar han er eller guidar han er? Vad är hans funktion i gruppen?**

JN:  
Det är mer guidning tycker jag, och sen har han ju koll på utvecklingen i sprinten och fångar upp om vi halkar efter eller ligger före. Men han trycker ofta på om vissa uppgifter i sprinten är extra viktiga, och att någon måste ta tag i det under dagen eller senast imorgon så att vi säkert hinner med det innan sprintens slut, så det är mer på den nivån. Och sen funkade han även som en av dem som har mest erfarenhet av att koda i det här systemet så det är ju väldigt bra att ha en person att kunna fråga.

**Intervjuare:**  
**Men då frågar ni honom som gruppmedlem, det är inte som ScrumMaster?**

JN:  
Ja det är sant, det är inte som ScrumMaster.

**Intervjuare:**  
**Gör ni en klar skillnad där, om ni kommer till honom som gruppmedlem eller som ScrumMaster?**

JN:  
Det flyter väl ihop lite, det gör det ju. Från och med den här sprinten så har vi delat upp oss i två team och han är ju ScrumMaster för båda, så han är faktiskt inte medlem i vårt team nu, så det är nog lite skillnad, det betyder att han inte jobbar aktivt med dem uppgifterna som vi har i vår sprint, tidigare har han varit med och gjort allting, så det är en ny skillnad, han får lite mer koordinatorrollen nu mot oss, och mindre teammedlem.

**Intervjuare:**  
**Så linjerna är lite klarare nu?**

JN:  
Ja, sen är det ju så att man fortfarande får fråga honom om det är kod-issues man undrar över eller så, men det är ändå klarare.

**Intervjuare:**  
**När frågar ni honom om råd angående Scrum? Kan du ge exempel på det?**

JN:  
Ett typiskt exempel är väl om en användare har en felrapport som inte vi kan återskapa, och man efter kontakt med användare känner att man inte når fram, och man fortfarande inte kan återskapa det, vad ska vi då göra? Då kan han vara lämplig att fråga vad han tycker om det där, och om vi ska kolla mer på utvärderingen. Ska vi jobba vidare, eller ska vi släppa det och ta de andra grejerna?

**Intervjuare:**  
**Hur tycker du det fungerar att planera en sprint i taget?**

JN:

Till skillnad från att planera flera sprints?

**Intervjuare:**

**Ja, eller att planera en längre tid. Nu är det ju bara två veckor.**

JN:

På ett sätt så känns det ju rätt så bra, man vet att nu de här två veckorna är det detta som gäller, å andra sidan så ställer det ju krav på produktägaren inför sprintplaneringen att se till att man är i linje med de långsiktiga målen också, så att det inte bara är det dagliga fixandet, framförallt nu när vi haft mycket buggrapporterad utveckling de sista sprintarna, men om man har mer nyutveckling så gäller det att man kan dela ut det på ett bra sätt så att det passar in i tvåveckorscyklerna. Och få in det i flera etapper.

**Intervjuare:**

**Men det känns inte som att din framtid är osäker för att man inte vet man vad man gör om två veckor?**

JN:

Nej det tycker jag inte, man vet ju inte exakt vilka grejer man kommer att jobba med, men det känns inte som att det är några problem med det. Även om man har planerat för längre tid så kan det ändå alltid hända saker som gör att man byter ut och gör annat istället. Det känns inte som en extra osäkerhet.

**Intervjuare:**

**Upplever du några svårigheter kopplade till samarbetet med Product Owner idag?**

JN:

Nej, det tycker jag inte, det vanliga kanske inte är att han sitter så nära oss som han gör i det här fallet, han har ju dessutom jobbat tidigare som utvecklare, så det kanske blir lite annorlunda.

**Intervjuare:**

**Stör han er under sprinten så att säga?**

JN:

Nej, det var kanske mer av det första sprinten, som han ofta kom med grejer, men nu är han rätt så inkörd på det här att nu är det sprinten som gäller, och att det är det som man får koncentrera sig på, så det tycker jag inte.

**Intervjuare:**

**Är det någonting som blivit bättre eller?**

JN:

Jo, precis. Det var ju inget stort problem i början heller, men det är väl ännu mindre problem nu så att säga.

**Intervjuare:**

**Och ni har morgonmöte varje dag?**

JN:

Japp, varje dag morgon klockan kvart över nio.

**Intervjuare:**

**Vad händer om man kommer för sent?**

JN:

Man blir väldigt pikad. Vi har snackat om att införa en bestraffning, men vi kom fram till att man får tänka på att komma i tid, och så ser vi om det fungerar. Men det är ju ändå viktigt att man håller tiden, att det inte är ok att komma in fem minuter efter, för det är ju ett störande moment.

**Intervjuare:**

**Schwaber, när han pratar om Scrum, så har han som guideline två dollar som böter, så kan man sen ge de pengarna till välgörande ändamål i slutet av varje sprint.**

JN:

Ja, det är ingen dum ide.

**Intervjuare:**

**Är det samma sak med Sprint Planning Meetings?**

JN:

Du menar om man kommer för sent? Ja, det är det väl, de har vi ju inte varje dag, men det är ju prioriterat att man är med. Idag hade vi ett möte där det var tre stycken som var med, utan sex stycken, och då kände vi att det var en miss, det var dumt, det hade varit bättre att skjuta på det mötet en dag så att det satt fler, för det påverkar commitment i gruppen. Så de mötena är ju viktiga, och alla ska helst vara med på dem. Men det är ännu mer på de här mötena man har varje morgon, om det alltid vore någon som kom för sent vore det ju inte så kul.

**Intervjuare:**

**Vad händer om ni inte hinner avklara allt som ska göras under en sprint?**

JN:

Ja det har vi pratat om, att det händer ju inte så himla mycket, vi har ju ett retrospective efter sprinten, så där tar vi ju upp hur vi lyckades med den här sprinten, var det saker vi hann med och saker vi inte hann med, och det är väl sagt att vi ska ligga inom plus minus 20 %, så att vi ska åtminstone hinna med 80 % av det vi sagt, annars är det ett tecken på att nånting inte fungerar som det ska. Men nu gjorde vi ju sämre än så på de första två sprintarna, vad vi sa då att vi följde upp vad som gick fel, hade vi misslyckats i planeringen eller är det något inom teamet som måste bli bättre? Så det är det som har hänt, att det har blivit högre attention på att vi ligger i fas. Vi har kanske även skärpt oss vad gäller planeringen och inte vara alltför tidsoptimistiska.

**Intervjuare:**

**Så det är estimaten som ni har blivit bättre på då?**

JN:

Ja, förhoppningsvis är det både estimaten som har blivit bättre och gruppen som presterar bättre också.

**Intervjuare:**

**En kombination alltså?**

JN:

Jag tror det.

**Intervjuare:**

**Har ni tillräckliga verktyg för att hjälpa till så att säga Scrumprocessen?**

JN:

Ja, vi kör det här ScrumWorks, något gratisverktyg, och det är inte perfekt, men jag tycker det fungerar tillräckligt för behovet. Dels så ser man vem som jobbar med vad i verktyget, det är ju rätt så grundläggande. Man ser även förändringen dag för dag hur vi ligger till i tiden.

**Intervjuare:**

**Sprint Backlog och så?**

JN:

Ja precis, och utifrån Sprint Backlog skapas då en sprint. Det är även i de här taskarna man lägger upp som man fyller i om man till exempel gjort en implement-task, så fyller man i att det finns en test- och en code review-task, som någon annan ska göra, och då fyller man i de här test-casen som ska köras, och de är de här filerna som ska Code Reviewas. Så det är ett ganska enkelt sätt att hålla ihop alla såna grejer.

**Intervjuare:**

**Använder du mycket Code Review?**

JN:

Ja det gör vi, vi Code Reviewar allting, och då gör vi även så att när vi ändrat i en klass så är det inte bara ändringen som ska reviewas, utan då ska hela klassen Code Reviewas, för i början av utvecklingen körde vi inte så mycket Code Review, så nu är tanken att vi ska gå igenom allting successivt. Det är Anders som drivit det här också, och det var i samband med att vi införde Scrumprocessen. Vi kände att det var ett bra sätt. Dels att man hjälper varandra, och att det höjer kvaliteten på koden.

**Intervjuare:**

**När ni började, hade ni något att säga till om vem som var med i teamet eller inte? I uppsättningen alltså?**

JN:

Nej, det tror jag inte, utan det bestämdes av linjeföraren, kanske med produktägaren, jag var inte inblandad.

**Intervjuare:**

**Men du känner att det är ett team med bra samlade kunskaper?**

JN:

Ja, det tycker jag. Nu har vi precis ny team-indelning med två team från och med imorgon, så varje team har nu lite mindre kunskap än vad det större teamet hade innan, men det känns ändå som att vi har tillräckligt för att kunna utföra arbetet.

**Intervjuare:**

**Ser du teamet som en ganska "homogen" grupp? Har ni liknande åsikter?**

JN:

Ja, rätt så homogen tycker jag, det är inte alla som tänker likadant, men de flesta tänker rätt så likartat när det gäller kodandet.

**Intervjuare:**

**Men ni har liknande åsikter över hur arbetet ska genomföras?**

JN:

Ja precis, absolut.

**Intervjuare:**

**Är det någon som är mer drivande i själva utvecklingsteamet, och har en mer tydlig roll?**

JN:

Just i teamet vi har nu är vi bara fyra personer och det känns som att det är utspritt rätt så likvärdigt. Jag tycker att Lars var rätt så drivande som teammedlem i det förra teamet men han hade ju samtidigt ScrumMaster-rollen så det ligger kanske lite i den rollen också att man ska vara lite mer drivande.

**Intervjuare:**

**Hur sker organiseringen inom teamet? Har ni interna roller som ni själva arbetat fram?**

JN:

Ja, till viss del. Vissa områden är det ju vissa som arbetat med en hel del, så på det viset blir det ju troligare att den personen, är det något med flashning så är det en person som tar det, men om det är något med power consumption så är det en annan. Men samtidigt så är det väldigt flytande, det beror ju på. Så det känns ändå som att alla är möjliga på alla roller.

**Intervjuare:**

**Kan du ge exempel på beslut som teamet fattat tillsammans? Och är det hela teamet som samarbetat eller är det en eller två som fattat beslut?**

JN:

Det är mer att två stycken tar diskussionen, iallafall när det gäller frågor som rör ett visst område, typ "är det ok

om jag inför den här lösningen”, då brukar vi inte involvera hela teamet. Ju mer det rör saker som berör Scrum, då är det självklart att vi har med hela teamet.

**Intervjuare:**

**Hur tycker du att kunskapsspridningen fungerar, du sa att vissa tar på sig en viss roll, har ni en viss kunskapsspridning?**

JN:

Ja, viss kunskapsspridning är det ju hela tiden, men det är ju fortfarande så att vissa kan vissa delar bäst. Men jag tror att vi försöker tänka på det, att det inte är endast en person som ska veta allt om ett visst område, det kan ställa till problem om det uppstår akuta fel och den personen inte är på plats.

**Intervjuare:**

**Vilka är enligt dig de största fördelarna med Scrum?**

JN:

Jag tycker att det främjar verkligen det här ”hjälpas åt”-konceptet., vilket ju inte betyder att man sitter två stycken bredvid varandra och parprogrammerar, men att modellen uppmuntrar att man bollar idéer, typ ”Vad tycker du om det här felet, jag tycker att man kan lösa det på det här viset, vad tror du om det?”. Så på det viset uppmuntrar det samarbete, absolut.

**Intervjuare:**

**Känner du dig mer motiverad att arbeta nu med Scrum-projektet än tidigare?**

JN:

Jag tror det, iallafall om man ska jämföra med att vara ensam utvecklare i ett stort projekt som ska vara klart om ett år, då är det ju roligare att det är mer feedback hela tiden på hur man ligger till, i och med att det är så korta cykler.

**Intervjuare:**

**Tror du att det hade fungerat sämre om man arbetat med en månads sprinter?**

JN:

Det tror jag också hade funkade bra faktiskt, men om det är optimalt med två, tre eller fyra veckor, det vet jag inte, jag tror inte det hade varit några problem med en månad heller. Fast två månader hade nog varit för lång tid.

**Intervjuare:**

**Kan du ge något exempel på grejer som förändrats i arbetet efter ett Sprint Retrospective?**

JN:

Det är nog att vi blir bättre och bättre på att bolla idéer med varandra och att be om hjälp av varandra. Det är det som är den största erfarenheten.

**Intervjuare:**

**Finner du någonsin någonting frustrerande med att arbeta med Scrum eller med agila tekniker?**

JN:

Det kan vara frustrerande mot slutet av en sprint att upptäcka ooh, nu är jag mitt uppe i det här, och hade behövt kanske en dag till för att riktigt gå i mål och göra det riktigt bra, men måste leverera någonting, så då står man med valet antingen backa, eller så satsar vi hårt och riskerar att skära bort någonting när vi ska leverera. Men det vi stryker kanske inte blir gjort förrän två tre sprintar framåt. Det kan vara frustrerande ibland, men det ingår i modellen att man ska komma till avslut.

**Intervjuare:**

**Jobbar ni övertid?**

JN:

Vi har inte gjort det särskilt mycket hittills.



**Intervjuare:**  
**Vem har tagit det beslutet?**

JN:

Ja.. Det är ju upp till varje enskild utvecklare, det är vissa som jobbat övertid, jag har inte gjort det.

**Intervjuare:**  
**Har du märkt några konflikter i ert Scrumteam? Det behöver inte handla om allvarliga konflikter.**

JN:

Det har ju varit diskussioner, givetvis har vi haft det, men det har ju inte varit några stora konflikter.

**Intervjuare:**  
**Yrkesrelaterade konflikter då eller? Typ "hur löser vi det här?"**

JN:

Ja, precis, jag skulle kalla det mer diskussioner än konflikter.

**Intervjuare:**  
**Vad tror du det inte blivit det?**

JN:

Det är en bra fråga, men jag tror det är för att vi tänker någorlunda lika, då är det lättare att man förstår varandra.

**Intervjuare:**  
**Har du sett några skillnader i teamets sätt att arbeta som du vill tillskriva Scrum, förutom som du nämnt tidigare med öppen kommunikation?**

JN:

Jag tror att det blir lite mer commitment, att man känner lite mer ansvar, nu har man ju faktiskt sagt att man ska hinna med allt, det är inte någon annan som bestämt och sagt att det här måste vara klart. Vi har själva sagt att "detta klarar vi på denna tiden", och det är faktiskt lite motiverande, "nu ska vi göra något bra av detta".

**Intervjuare:**  
**Finns det något du vill ändra i Scrumprocessen?**

JN:

Nej, ingen speciell punkt som jag känner att jag vill ändra.

**Intervjuare:**  
**Har ni någonsin avbrutit en sprint?**

JN:

Nej, det har vi inte gjort.

**Intervjuare:**  
**Men ni har rätt att göra det?**

JN:

Ja, det kan vi göra, och produktägaren kan säga att nu avbryter vi den här sprinten och planerar om. Det kanske har tillkommit nya saker som måste in, eller att någonting tar för lång tid.

**Intervjuare:**  
**Upplever du att produktägaren, du sa innan att hans arbete blivit bättre och bättre sedan första sprinten, tror du det hade varit annorlunda om han hade suttit i en annan lokal? Tror du det hade varit negativt eller positivt?**

JN:

Just nu tror jag att det är en fördel för oss att han sitter nära, för det är ofta vi måste gå till honom under sprintarna, t.ex. ”Hur gör ni nu med den här DMSen, vi har kommit fram till att det är två delar, den ena kan vi lösa ganska enkelt, medan den andra är lite besvärlig”.

**Intervjuare:**

**Vad är DMS?**

JN:

DMS är en felrapport helt enkelt, så som det är nu med att vi rättar buggar så är det DMSarna som våra Backlog Items baseras på.

**Intervjuare:**

**Japp, suveränt. Då har vi inte så mycket mer. Tack så hemskt mycket!**

JN:

Tack själva.

## Bilaga 17, Exempel på Unit Test

```
/// <summary>
/// Tests the CalculateReportedAmmount
/// </summary>
[Test]
public void TestCalculateReportedAmmount()
{
    //Create the Activity and set the price to 950 / hour.
    Activity m_activity = new Activity();
    m_activity.PricePerHour = 950;

    //Create two WorkSheetItem
    IWorkSheetItem i1 = m_mock.Stub<IWorkSheetItem>();
    i1.Activity = m_activity;
    i1.TimeWorked = 23F;

    IWorkSheetItem i2 = m_mock.Stub<IWorkSheetItem>();
    i2.Activity = m_activity;
    i2.TimeWorked = 24.5F;

    //Add the WorkSheetItem to the Activity
    m_activity.WorkSheetItems.Add(i1);
    m_activity.WorkSheetItems.Add(i2);

    //The CalculateReportedAmmount takes the ammount of hours * the price
    //per hour
    Assert.AreEqual(m_activity.CalculateReportedAmmount(), 45125);
}
```

## Bilaga 18, Exempel på Use Case

**Title:** Lista ej levererade beställningar

**Description:** Tillåter en inloggad användare med administratörsrättigheter att se en lista med alla beställningar som ännu inte levererats till beställare.

**Primary Actor:** Administratör

**Success Guarantee:** Alla ej levererade beställningar listas

**Preconditions:** Användaren måste vara inloggad som administratör

**Triggers:** Användaren efterfrågar ej levererade beställningar i administrationsgränssnittet

**Main Success Scenario:**

1. Användaren klickar på Beställningsinformation i menyn
2. Användaren klickar på fliken för ej levererade beställningar
3. Användaren väljer datumintervall i rullgardinsmenyn
4. Användaren klickar på knappen för att lista ej levererade beställningar
5. Användaren får nu se alla beställningar som stämmer in på datumintervallet

**Extensions:**

1a. Användaren har inte blivit tilldelad rollen beställningsadministratör, och blir då upplyst om att han saknar behörighet.

**Notes:**

En beställning räknas som ej levererad tills den paketerats och lämnats till postombud.

## **Bilaga 19, Exempel på User Story**

Lista ej levererade beställningar

Användaren får se en lista över ej levererade beställningar inom ett datumintervall.