
LUNDS UNIVERSITET
Ekonomihögskolan

*Institutionen för informatik*

# DevOps team – vad ska vara med?

## En kvalitativ studie om vilka mjuka- och hårda förmågor organisationer tar i beaktning vid skapandet av DevOps team.

Kandidatuppsats 15 hp, kurs SYSK16 i Informatik

Författare: Matilda Haglund
Ulrika Lindstrand
Nina Micin

Handledare: Magnus Wärja

Rättande lärare: Christina Keller
Markus Lahtinen

# DevOps team – vad ska vara med? : En kvalitativ studie om vilka mjuka- och hårda förmågor organisationer tar i beaktning vid skapandet av DevOps team.

ENGELSK TITEL: DevOps team – What to Include? : A qualitative study of soft- and hard skills that organizations take into consideration when creating DevOps teams.

FÖRFATTARE: Matilda Haglund, Ulrika Lindstrand och Nina Micin

UTGIVARE: Institutionen för informatik, Ekonomihögskolan, Lunds universitet

EXAMINATOR: Odd Steen, Docent, Fil Dr

FRAMLAGD: maj, 2019

DOKUMENTTYP: Kandidatuppsats

ANTAL SIDOR: 91

NYCKELORD: DevOps, mjuka förmågor, hårda förmågor, förmågor, teamstruktur, team.

SAMMANFATTNING (MAX. 200 ORD):
Den konstanta förändringen sker ständigt i dagens snabbt föränderliga och konkurrensutsatta IT-värld. För att hänga med i förändringarna kommer organisationer behöva anamma DevOps. DevOps är en sammanslagning av de två orden Development (Dev) och Operations (Ops) och gör det möjligt för organisationer att snabbt och frekvent kunna leverera mjukvara av hög kvalitet till kund. När organisationer implementerar DevOps skapar de oftast DevOps team. Medlemmarna i dessa team bör besitta vissa mjuka och hårda förmågor för att kunna göra snabba och frekventa leveranser. Syftet med studien är att påvisa vilka mjuka- och hårda förmågor hos individen organisationer tar i beaktning vid skapande av DevOps team. Vi genomför därför intervjuer med företag som har skapat DevOps team. Våra resultat visar att organisationer är eniga kring fem förmågor som är viktiga att beakta. Dessa är anpassningsbarhet, beslutsfattande, öppenhet för förändring, kunskapsdelning och kommunikation.

# Innehåll

# Figurer



# Tabeller

# 1 Inledning

## 1.1 Bakgrund

Den konstanta förändringen sker ständigt snabbare i dagens föränderliga och konkurrensutsatta IT-värld. I dagens utvecklingsföretag arbetar utvecklingsavdelningar ofta agilt, det vill säga att de jobbar i cykler för att snabbt producera kod för att kunna behålla en konkurrenskraftig position på marknaden genom att möta kundernas ständigt skiftande krav (Ingdahl, 2016). Den agila miljön har bidragit till en flexibel utvecklingsprocess men trots detta uppstår flaskhalsar när koden ska produktionssättas (Debois, 2011). Enligt Debois (2011) beror detta delvis på att utvecklare och den operativa verksamheten använder olika miljöer där kod från utvecklarna inte fungerar i den operativa verksamhetens miljö. Även för att arbetsuppgifterna ligger förlagda på olika avdelningar med helt skilda mål. Den operativa verksamheten är ansvariga för att sätta mjukvara i produktionsmiljö så att mjukvara kan göras tillgänglig för kund och för underhåll av mjukvara (Hüttermann, 2012). Följaktligen uppstår det en barriär mellan avdelningarna och samarbetet är mer eller mindre obefintligt, vilket förhindrar snabba och kontinuerliga uppdateringar till kund och resulterar i förlorat värde för organisationer (Lwakatare, Kuvaja, & Oivo, 2015).

DevOps är en sammanslagning av de två orden Development (Dev) och Operations (Ops) (Debois, 2011). Konceptet myntades under en konferens år 2009 av Patrick Debois som förespråkade ett större samarbete för att lösa den rådande problematiken mellan avdelningarna (Paul, 2014). Erich (2018) definierar DevOps som ett samarbete mellan utvecklare och den operativa verksamheten och menar att hur organisationer väljer att etablera samarbetet är upp till var och en. Det vanligaste är däremot att nya team skapas eller nuvarande teams arbetsuppgifter förändras (Erich, 2018; Shahin, Zahedi, Babar, & Zhu, 2017). I denna studie definieras dessa team som DevOps team.

Ett team är två eller fler individer som har etablerat ett gemensamt mål, procedurer, har förmågor som kompletterar varandra, är beroende av varandra och kan anpassa sig för att producera resultat som alla tar ansvar för och som uppnår de uppsatta målen (Katzenbach & Smith, 2005; Kinlaw & Liungman, 1995; Morgan Jr, Glickman, Woodard, Blaiwes, & Salas, 1986; Wheelan, 2017). Det finns flertalet faktorer man kan ta i beaktning när man skapar team, där förmågor hos medlemmar har påvisats vara en av de viktigare (Frostenson, 1990; Harrison, Price, Gavin, & Florey, 2002; Katzenbach & Smith, 2005; Wheelan, 2017).

Agila team ligger som grund till DevOps team då DevOps är en förlängning av det agila arbetssättet (Ebert, Gallardo, Hernantes, & Serrano, 2016). Båda teamen jobbar för att effektivisera processen och snabbare få ut produkten till kunden samt kräver båda ett effektivt samarbete och kommunikation (Overhage & Schlauderer, 2012; Wiedemann & Wiesche, 2018). Trots sina likheter skiljer sig ett DevOps team från ett agilt team på flera olika sätt. Kim, Humble, Debois, och Willis (2016) beskriver några positiva effekter av DevOps gentemot det agila arbetssätt. De pekar på att i ett DevOps team kan kod produceras effektivare och mer frekvent, samt att det är lättare att hitta fel i koden med hjälp av automatiserade tester. Detta gör enligt Kim et al. (2016) att utvecklare lättare kan lära sig av

sina misstag och minska mängden teknisk skuld då de kan ta sig an misstaget direkt. I slutändan säger Kim et al. (2016) att arbetarna i ett DevOps team känner sig mer produktiva då de inte behöver vänta på varandra i olika processer. Medarbetarna är inkluderade under hela utvecklingsprocessen och har därmed uppsikt över hela förloppet.

## 1.2 Problemområde

Sebastian, Ross, Beath, Mocker, Moloney, och Fonstad (2017) menar att det kommer bli nödvändigt för organisationer att anamma DevOps för att kunna vara konkurrenskraftiga på marknaden. Därför har tidigare studier identifierat vilka förmågor och kapabiliteter DevOps team borde ha för att snabbt leverera tjänster och produkter till kund och därmed generera konkurrensfördelar till organisationer (Hemon, Lyonnet, Rowe, & Fitzgerald, 2019; Wiedemann & Wiesche, 2018; Wiedemann & Schulz, 2017).

Förmågor går att dela in i mjuka- och hårda förmågor (Robles, 2012; Gallivan, Truex III, & Kvasny, 2004). Mjuka förmågor i Robles (2012) mening är interpersonella förmågor och personliga kvaliteter. Interpersonella förmågor beskriver vilken relation en individ har till andra medan personliga kvaliteter syftar mer på en människas personlighet, hur omtyckt samt hur strukturerad en person är (Robles, 2012). Några exempel på mjuka förmågor är viljan att lära sig, kunskapsdelning och kommunikation. Hårda förmågor till skillnad från mjuka förmågor är när en individ behärskar utförandet av en specifik aktivitet genom att följa procedurer samt använda verktyg och tekniker (Katz, 2009). Några exempel på hårda förmågor är kunskap om verktyg för DevOps, testning och kontinuerliga aktiviteter. Stevens och Norman (2016) visade med sin studie att arbetsgivare lägger större vikt vid mjuka förmågor än hårda förmågor när de rekryterar nyexaminerade IT studenter då mjuka förmågor är svårare att lära sig. Enligt Lobosco (2019) säger 92% av rekryterare att mjuka förmågor har lika stor vikt som hårda förmågor eller att de till och med väger tyngre än dessa. Att mjuka förmågor har fått en större vikt ligger till grund i att flera arbetsprocesser som kräver hårda förmågor automatiseras och därför lägger företag ett större fokus på mjuka förmågor då de inte kan automatiseras (Lobosco, 2019).

Studierna undersökte däremot inte vilka förmågor hos individen som i praktiken tas i beaktning vid skapande av DevOps team. Fitzgerald och Stol (2014) bekräftar att hur organisationer sätter ihop DevOps team är ett outforskat område. Sebastian et al. (2017) menar att det kommer bli nödvändigt för organisationer att anamma DevOps för att förbli konkurrenskraftiga. För att som individ bli utvald till att ingå i ett DevOps team kan det komma att bli nödvändigt att besitta de förmågor som värderas högst bland organisationer när de väljer ut teammedlemmar. Genom att studera vilka mjuka- och hårda förmågor hos individen organisationer tar i beaktning när de skapar DevOps team, kan resultatet användas som underlag vid bildandet av framtida team. Detta för att öka möjligheterna att skapa ett välfungerande team för att uppnå den fulla effekten och värdet av att implementera DevOps. Universitet och utbildningsväsendet kan även använda sig av denna studie för att veta vilka förmågor som arbetsgivare anser är viktiga och därefter välja att fokusera på dessa i ett utbildningssyfte. Detta leder oss till vår forskningsfråga:

*Vilka mjuka- och hårda förmågor hos individen tar organisationer i beaktning när de skapar DevOps team?*

## 1.3 Syfte

Syftet med studien är att påvisa vilka mjuka- och hårda förmågor hos individen organisationer tar i beaktning vid skapande av DevOps team och jämföra de med förmågorna vi har identifierat i litteraturen.

# 2 Litteraturgenomgång

*I detta kapitel presenteras den teori som ligger till grund för att besvara frågeställningen. Vi börjar med att definiera begreppet DevOps samt DevOps team. Vidare definierar vi begreppet team samt redogör för vilka aspekter man kan ta hänsyn till när man skapar team och påvisar vikten av att ta individens förmågor i beaktning. Därefter definierar vi förmåga och redogör för två typer av förmågor. Slutligen delar vi in tidigare identifierade förmågor DevOps team bör ha i de två presenterade typerna.*

## 2.1 DevOps

### *2.1.1 Definition*

Erich, Amrit, och Daneva (2014) menar att DevOps saknar en enhetlig definition där vissa definierar det som ett ramverk, en jobbtitel eller som en uppsättning förmågor. Därför genomförde Erich (2018) en systematisk litteraturstudie och kvalitativ studie för att utforma en definition baserat ur ett akademiker- och praktikers perspektiv. Erich (2018) kom fram till att DevOps handlar om samarbete mellan systemutvecklare och den operativa verksamheten. Vidare menar Erich (2018) att sättet organisationer etablerar samarbetet varierar då varje organisation själva väljer hur de ska göra. För denna studie kommer vi använda definitionen Erich (2018) tagit fram.

Trots att Erich (2018) menar att ett av de mest omnämnda ramverken som hittills har föreslagits för att definiera DevOps mynnar ut i hennes definition, tycker vi att det är av betydelse att skapa en förståelse för vilka principer som har föreslagits att ingå i DevOps. Detta eftersom de förmågor för DevOps team som tidigare studier identifierat och som vi kommer utgå ifrån i vår studie bygger på principerna. Ramverket går under benämningen CAMS (Erich, 2018).

Enligt Perera, Silva, och Perera (2017) består DevOps av fyra kärnvärden som benämns CAMS. CAMS står för culture, automation, measuring och sharing (Perera et al., 2017). Vi kommer hädanefter benämna principerna som kultur, automation, mätning och delning.

*Kultur* - Organisationer behöver skapa en kultur som möjliggör ett ökat samarbete mellan systemutvecklare och den operativa verksamheten samt för att se till att de anställda accepterar de förändringar DevOps medför (Erich, 2018; Perera et al., 2017). Shamow (2011) menar att organisationer behöver skapa en kultur där allt ska finnas öppet samt tillgängligt och där man ger och tar. För att lyckas med denna kulturella förändring krävs det ett genuint engagemang, dels från den eller de som driver förändringen och dels från de personer som kommer att delta i den (Shamow, 2011).

*Automation* - Brunnert et al. (2015) menar att för att lyckas med en implementering av DevOps krävs automatisering av både utvecklings- och förvaltningsprocessen. I

förvaltningsprocessen automatiseras bland annat placering av kod i produktionsmiljö och konfigurering av kod medan bland annat tester automatiseras i utvecklingsprocessen (Lwakatare et al., 2015). Hüttermann (2012) menar att automation bidrar till att minska utvecklingstiden och är en väsentlig del för att kunna få snabb feedback. Automatisering är även en viktig framgångsfaktor när det kommer till ett effektivt samarbete mellan utvecklare och den operativa verksamheten (Wettinger, Breitenbücher, & Leymann, 2014).

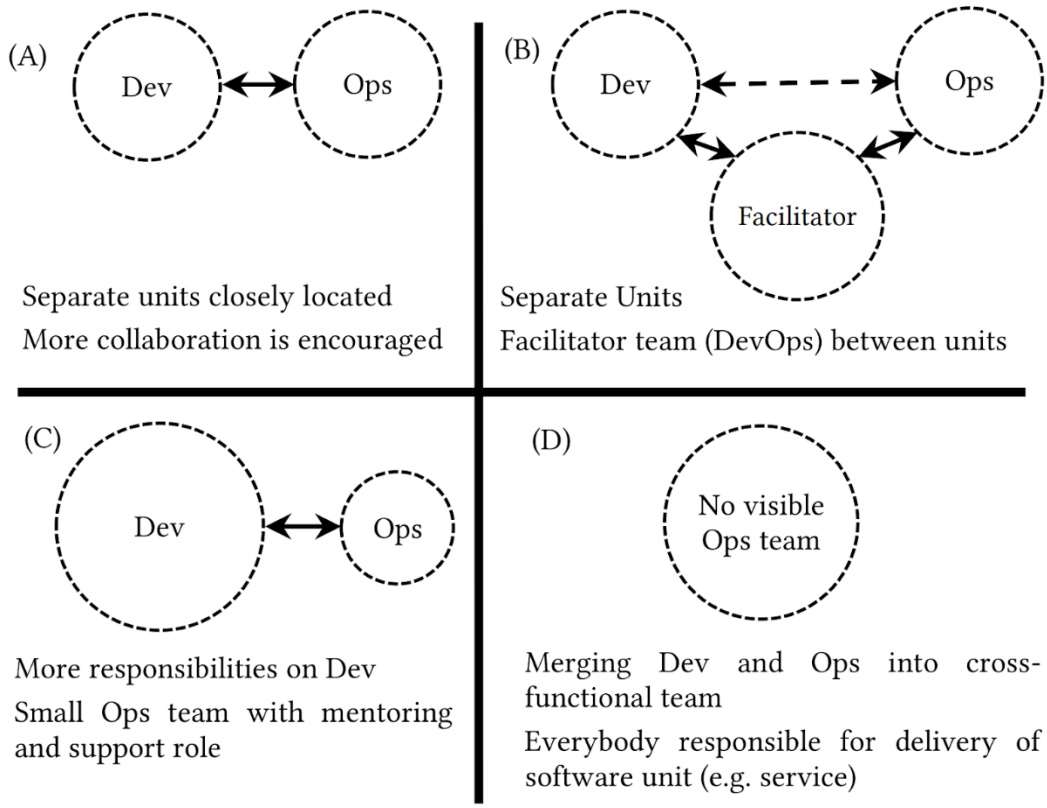
*Mätning* - Hüttermann (2012) menar att mätning är en avgörande aspekt för att identifiera sin nuvarande status och för att veta om man är på rätt väg. Mätning innebär att man identifierar gemensamma mätvärden för både systemutvecklare och den operativa verksamheten för att undvika att sträva efter olika resultat (Hemon et al., 2019; Shropshire, Menard, & Sweeney, 2017). Lwakatare et al. (2015) pekar på att även resultaten av utvecklingsprocessen bör mätas i form av användning och prestanda av funktionalitet.

*Delning* - Att dela med sig av idéer, lärdomar, verktyg och ta gemensamt ansvar för problem är en förutsättning för att öka samarbetet mellan utvecklare och den operativa verksamheten (Perera et al., 2017).

### *2.1.2 DevOps team*

DevOps team ser olika ut från organisation till organisation (Erich, 2018; Shahin et al., 2017). Flertalet studier menar att DevOps förespråkar skapandet av tvärfunktionella team som har ansvar för alla aktiviteter relaterade till en produkt eller tjänst (Balalaie, Heydarnoori, & Jamshidi, 2016; Debois, 2011; Ebert et al., 2016; Fitzgerald & Stol, 2014). Däremot visar studier av Shahin et al. (2017) och Erich (2018) att organisationer väljer att använda olika teamstrukturer vid användning DevOps. Shahin et al. (2017) beskriver teamstrukturer i ett DevOps sammanhang som olika sätt en verksamhet kan organisera utvecklingsavdelningen och den operativa verksamheten för att öka samarbetet mellan dem.

Shahin et al. (2017) identifierade fyra teamstrukturer organisationer använder. Organisationer låter antingen systemutveckling och den operativa verksamheten förbli separata avdelningar som placeras fysiskt närmare varandra och större vikt läggs vid samarbete. Vidare kan de skapa så kallade *DevOps team* som agerar förmedlare mellan utvecklingsteam och team från den operativa verksamheten. De kan också ge utvecklingsteam ansvar för att driftsätta kod med stöd av mindre team från den operativa verksamheten som sköter uppgifter såsom övervakning av system. Slutligen visar Shahin et al. (2017) att vissa organisationer skapar tvärfunktionella team där både systemutvecklare och anställda från den operativa verksamheten ingår och som ansvarar för en hel tjänst eller produkt.




**Figur 1:** Teamstrukturer (Shahin et al., 2017 )

I den här studien kommer vi att studera teamstrukturer där antingen nya team skapats eller där befintliga team har förändrats såsom deras arbetsuppgifter, vilket enligt Figur 1 är strukturerna B, C och D. Dessa team kommer vi benämna *DevOps team*. För att skapa en större förståelse för vad som karaktäriserar ett team kommer vi definiera begreppet.

## 2.2 Team

Team kan beskrivas som en grupp människor som tillsammans arbetar för att lösa en arbetsuppgift och där medlemmarna får möjlighet att använda sig av och dela med sig av sin kunskap (Kinlaw & Liungman, 1995). Katzenbach och Smith (2005) samt Wheelan (2017) beskriver team i mer detalj som en liten grupp människor som har etablerat ett gemensamt syfte, mål och arbetsprocedurer samt har förmågor som kompletterar varandra för att producera ett gemensamt resultat som alla tar ansvar för. Morgan Jr et al. (1986) specificerar ett team ytterligare genom att definiera det som två eller fler individer som är beroende av varandra för att nå gemensamma mål och som kan anpassa sig för att nå målen.

Vi kommer beskriva team som två eller flera individer som har etablerat ett gemensamt mål, procedurer, har förmågor som kompletterar varandra, är beroende av varandra och kan anpassa sig för att producera resultat som alla tar ansvar för och som uppnår de uppsatta målen. Definitionen kommer användas för att säkerställa att vår syn på team stämmer överens med den hos våra respondenter.

Vidare kommer vi redogöra för vilka olika faktorer organisationer kan ta i beaktning när de skapar team för att påvisa att författarna till den här studien är medvetna om att organisationer inte nödvändigtvis enbart ser till förmågor. Detta är även något som kan framkomma i den data vi samlar in. Däremot kommer vi påvisa vikten av att ta hänsyn till individers förmågor genom att ställa det i relation till de andra faktorerna som kan beaktas.

### *2.2.1 Skapa team*

Rubenowitz (2004) säger att vid sammansättning av team bör man sätta ihop individer med liknande intressen, attityder och värderingar eftersom dessa enligt studier jobbar bättre tillsammans än om de inte har det. Vidare menar Rubenowitz (2004) att sätta ihop individer som ogillar varandra i hopp om att de kommer lära sig arbeta tillsammans inte är en väg man bör gå. Däremot argumenterar Wheelan (2017) tvärtom för att teammedlemmar inte behöver tycka om varandra eller ha liknande intressen för att jobba effektivt ihop. Istället är det viktigare att medlemmar har nödvändig teknisk förmåga för att lösa uppgiften samt förmåga att arbeta i och förbättra team (Wheelan, 2017). Även Frostenson (1990) bekräftar Wheelans (2017) argumentation gällande medlemmarnas tekniska kunskaper genom att peka på att team sätts ihop efter de kunskaper som krävs vid ett specifikt uppdrag.

Katzenbach och Smith (2005) menar att organisationer ofta missar att ta i beaktning vilka förmågor som behövs till uppdraget innan team sätts ihop och att man bör välja individer baserat på deras förmågor och inte personlighet. Förmågor växer fram i team men vissa personer behöver ha dessa sedan tidigare (Katzenbach & Smith, 2005). Förmågorna Katzenbach och Smith (2005) beskriver innefattar tekniska, interpersonella, problemlösning och beslutsfattande förmågor.

Samtidigt menar Robles (2012) att interpersonella förmågor är en del av en individs personlighet. Harrison et al. (2002) identifierade att psykologisk mångfald har betydelse över tid för hur väl team presterar. Psykologisk mångfald är skillnader i teammedlemmars personlighet, förmågor och värderingar (Jackson, May, & Whitney, 1995). Dessa skillnader kommer fram ju mer man lär känna en individ och har större betydelse för ett teams prestation än demografisk mångfald över tid (Harrison et al., 2002) vilket också är en faktor som organisationer kan ta hänsyn till när de sätter ihop team (Hollenbeck, DeRue, & Guzzo, 2004). Jackson et al. (1995) definierar demografisk mångfald som attribut man snabbt kan identifiera vid interaktion med en individ såsom ålder, kön, etnicitet och religiös eller politisk tillhörighet.

Det kan även vara så att organisationer väljer teammedlemmar inom informella nätverk. Dessa nätverk består av vänner, familj, kontakter och slumpmässiga interaktioner med andra (Birley, 1985; Krackhardt & Stern, 1988).

För att besvara vilka mjuka- och hårda förmågor hos individer organisationer tar hänsyn till vid skapande av DevOps team är det av intresse att definiera begreppet förmåga.

## 2.3 Förmågor

Peterson och Van Fleet (2004) definierar en förmåga som möjligheten att utföra en specifik uppgift med hjälp av den information och kunskap man besitter inom ett område. Vidare

menar Katz (2009) att en förmåga är något man gör, något man kan bli bättre på, som går att mäta och som man nödvändigtvis inte är född med. Vi kommer utgå från båda dessa definitioner i denna studie. Robles (2012) samt Gallivan et al. (2004) pekar även på att förmågor går att dela in i mjuka förmågor och hårda förmågor. Vi kommer därför redogöra för definitionen av mjuka- och hårda förmågor.

Stevens och Norman (2016) visade med sin studie att arbetsgivare lägger större vikt vid mjuka förmågor än hårda förmågor när de rekryterar nyexaminerade IT studenter då mjuka förmågor är svårare att lära sig. Mjuka förmågor menar Schulz (2008) är något som är svårt att definiera då det kan tolkas olika från ett sammanhang till ett annat men att det utgör en betydande del i utformningen av en människas personlighet. Robles (2012) definitionen ligger i linje med Schulz (2008) då hon menar att mjuka förmågor fastställer en del av en människas personlighet och har mer att göra med vem man är än vilken kunskap man besitter. Vidare säger Robles (2012) att mjuka förmågor innefattar interpersonella förmågor, personliga kvaliteter och kvaliteter som är bra att ha i arbetslivet. Interpersonella förmågor handlar om hur bra en person är på att interagera med andra (Robles, 2012). Shakir (2009) inkluderar även förmågan att fatta beslut och lösa problem i interpersonella förmågor. Personliga kvaliteter innefattar personlighet, hur omtyckt samt hur strukturerad en person är. Kvaliteter för arbetslivet inkluderar kommunikativa- och ledarskapsförmågor samt hur väl en individ arbetar i team (Robles, 2012).

Hårda förmågor även kallat tekniska förmågor, är benämningen på förmågor som syftar på den tekniska expertisen och kunskapen som krävs för ett arbete (Robles, 2012). Vidare menar Katz (2009) att en teknisk förmåga är när en individ behärskar utförandet av en specifik aktivitet genom att följa procedurer samt använda verktyg och tekniker.

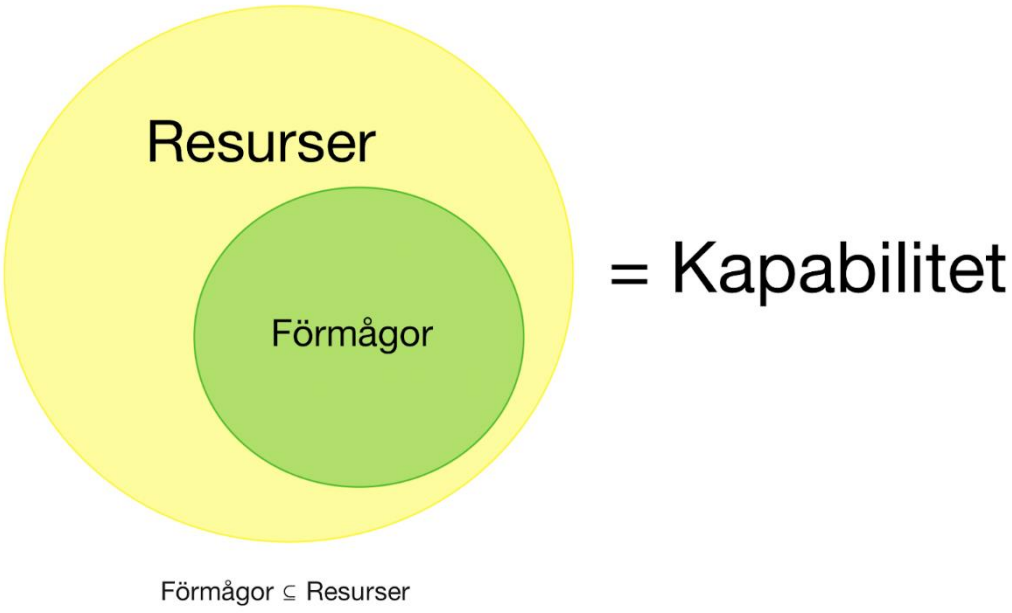
Vidare kommer vi redogöra för vilka förmågor DevOps team bör ha enligt litteraturen för att kunna besvara vilka mjuka- och hårda förmågor hos individer organisationer tar i beaktning vid skapande av DevOps team.

## 2.4 Förmågor för medlemmar i ett DevOps team

Såvitt vi har kunnat se finns det tre studier som identifierar förmågor samt kapabiliteter DevOps team bör ha. Dessa studier definierar DevOps team som tvärfunktionella team (Hemon et al., 2019; Wiedemann & Wiesche, 2018; Wiedemann & Schulz, 2017). Däremot menar Wiedemann och Wiesche (2018) att deras uppsättning förmågor gäller oavsett struktur i organisationer som anammar DevOps vilket gör deras resultat relevant för att besvara vår frågeställning. Wiedemann och Schulz (2017) samt Hemon et al. (2019) pekar inte på att deras resultat kan appliceras på andra teamstrukturer. Däremot menar de att kapabiliteterna respektive förmågorna möjliggör för DevOps team att snabbt leverera mjukvara till kund, vilket även är ett av målen med DevOps (Lwakatare et al., 2015). Därav kan alla DevOps team oavsett teamstruktur ha samma syfte, nämligen en snabb leverans till kund, vilket också motiverar oss att använda kapabiliteterna och förmågorna på andra teamstrukturer.

Kapabilitet definieras som en kombination av resurser som organisationer använder för att uppnå önskade resultat (Wiedemann & Schulz, 2017). Resurserna i sig kan ta sig i form av exempelvis förmågor hos teammedlemmar, teknologier, råmaterial och kunskap (Wiedemann & Schulz, 2017). Trots att Wiedemann och Schulz (2017) identifierar kapabiliteter och inte förmågor blir deras resultat relevant för vår studie. Anledningen är att vissa av kapabiliteterna

består av förmågor hos individen. För varje relevant kapabilitet kommer vi beskriva den eller de förmågor som krävs hos individen för att uppnå kapabiliteten. Dessa förmågor kommer sedan, tillsammans med identifierade förmågor från övrig litteratur, användas för att samla in empiri.




**Figur 2:** Illustrerar begreppet kapabilitet där en kapabilitet består av flera resurser. En resurs kan sedan vara en förmåga hos en individ men behöver nödvändigtvis inte vara det.

Wiedemann och Wiesche (2018) tog fram vad de kallar den ideala uppsättningen förmågor för DevOps team. De definierar förmåga som användning av de kunskaper och personliga kvaliteter en individ har för att lösa en uppgift samt för att bedöma hur individen ska agera i olika sammanhang (Wiedemann & Wiesche, 2018). Deras definition stämmer överens med Peterson och Van Fleet (2004) samt Katz (2009) gällande att det handlar om att applicera kunskap för att göra något. När de sedan nämner att även personliga kvaliteter kan användas så går det i linje med Robles (2012) definition av mjuka förmågor.

Hemon et al. (2019) beskriver vilka nya förmågor DevOps team förväntas ha gentemot agila team. Vidare menar författarna att en förmåga är när en individ kan utföra något bra med hjälp av den kunskap och erfarenhet de besitter samt att de kan bli bättre i sitt utförande. Deras definition går därför i linje med de definitioner denna studie använder sig av vilket även gör deras resultat relevant för att besvara vår frågeställning.

Utöver att redogöra för förmågor enligt litteraturen som DevOps team bör ha är det även intressant att kategorisera de identifierade förmågorna i mjuka- och hårda förmågor. Detta för att kunna besvara vilka mjuka- och hårda förmågor som är viktiga vid val av teammedlemmar.

### *2.4.1 Mjuka förmågor*

Wiedemann och Schulz (2017) identifierade villighet att förändras, på engelska *change readiness*, som en kapabilitet DevOps team bör ha. Vilja att förändras innebär att organisationer kan bemöta förändringar genom att snabbt leverera IT tjänster och produkter

(Clark, Cavanaugh, Brown, & Sambamurthy, 1997). Kapabiliteten blir viktig för DevOps team eftersom den uppmuntrar självstyrande, samarbete och kunskapsdelning inom teamet (Shahin et al., 2017; Wiedemann & Schulz, 2017). För att DevOps team ska kunna uppnå kapabiliteten behöver medlemmar enligt Wiedemann och Schulz (2017) kunna ta över arbetsuppgifter från varandra. Vi kommer hädanefter att benämna förmågan ***anpassningsbarhet***. För att möjliggöra det letar organisationer efter individer med en t-formad profil (Wiedemann & Schulz, 2017).

En individ med en t-formad profil är enligt Donofrio, Spohrer, Zadeh, och Demirkan (2010) en person som är specialist inom ett område men har även kunskap inom andra områden. Att kunna ta över arbetsuppgifter kan innebära att utföra en specifik uppgift med den information man besitter. Det är dessutom något man gör och något man skulle kunna bli bättre på. Därför stämmer den in på de definitioner av förmåga vi använder i studien och kommer därför ses som en förmåga. T-formade individer har en stark vilja att lära sig nya saker vilket också möjliggör för de att erhålla kunskap inom både sitt egna men också andra områden (Donofrio et al., 2010). Därför kan förmågan även kategoriseras som en mjuk förmåga.

DevOps team ska kunna ***ta snabba beslut själva*** och stå för konsekvenserna av besluten (Wiedemann & Wiesche, 2018; Wiedemann & Schulz, 2017). Detta blir viktigt för DevOps team eftersom de måste kunna ta beslut för att bemöta oväntade händelser relaterade till den tjänst eller produkt de ansvarar för. Dessa händelser kan inkludera förändringar på marknaden eller att tjänsten eller produkten ligger nere (Wiedemann & Wiesche, 2018; Wiedemann & Schulz, 2017). Eftersom förmågan att kunna ta beslut enligt Shakir (2009) är en mjuk förmåga kommer även vi kategorisera den som sådan.

För att kunna ta beslut menar Wiedemann och Schulz (2017) att teammedlemmar måste dela med sig av kunskap inom teamet och vara villiga att lära av varandra. Wiedemann och Wiesche (2018) och Hemon et al. (2019) identifierar ***kunskapsdelning***, ***vilja att lära sig*** nya saker av andra teammedlemmar och ***kommunikation*** som förmågor medlemmar i DevOps team bör ha när de ingår i teamet. Även Hussain, Clear, och MacDonell (2017) menar att anställda som jobbar enligt DevOps bör kunna dela med sig av kunskap, kommunicera och vara villiga att lära sig nya saker. Kommunikativa förmågor innefattar verbal och skriftlig kommunikation samt att kunna hålla presentationer och lyssna på andra (Hemon et al., 2019). Jelphs (2006) menar att en viktig grundläggande förmåga är just kommunikation och som man utgår från att de flesta besitter. Dock har avsaknaden av denna förmåga varit stor när det kommer till verkligheten och varit en bidragande orsak till många olyckor som drabbat företag i olika branscher under de senaste åren (Jelphs, 2006).

Kommunikation kategoriseras som en mjuk förmåga (Robles, 2012). Eftersom kommunikation är en förutsättning för kunskapsdelning (Melnik & Maurer, 2004; Riege, 2005) kommer vi även kategorisera kunskapsdelning som en mjuk förmåga. Det finns fler förutsättningar men för studiens syfte kommer vi fokusera på kommunikation eftersom det är den enda förutsättningen som är en förmåga hos individer (Riege, 2005).

Hur villig en individ är att lära sig kan förklaras med hjälp av personlighet och den miljö en individ befinner sig i (Major, Turner, & Fletcher, 2006). Däremot menar Major et al. (2006) att personlighet avgör vår benägenhet att agera på ett visst sätt och att miljö därför har olika inverkan på viljan att lära sig. Exempelvis, individer med en personlighet som gör de mer benägna att vilja lära sig behöver inte lika mycket stöd eller uppmuntran från sin miljö för att vilja lära sig som individer med mindre benägenhet (Major et al., 2006). Eftersom studiens

fokus ligger på förmågor och Robles (2012) argumenterar för att personlighet är en mjuk förmåga, kommer vi räkna vilja att lära sig som en mjuk förmåga.

Wiedemann och Schulz (2017) identifierar samarbetskultur inom DevOps team som en kapabilitet och menar med det att medlemmar i ett DevOps team ska kunna samarbeta. Enligt Hemon et al. (2019) förändras sättet individer samarbetar på inom en organisation när de anammar DevOps. Därför ska teammedlemmar kunna anamma det nya sättet att samarbeta på som DevOps förespråkar. För att göra det ska teammedlemmar vara ***öppna för förändringar*** (Wiedemann & Schulz, 2017) vilket även Hussain et al., (2017) och Hemon et al. (2019) anser att individer som jobbar med DevOps ska vara. Hur pass öppen en individ är för förändringar kan förklaras med hjälp av personlighetsdraget *Openness to Experience* som är en av fem drag av personlighet enligt The Big Five Model (Barrick & Mount, 1991). Personlighetsdraget tar sig i form av nyfikenhet, öppenhet och okonventionellt agerande (Barrick & Mount, 1991; Judge, Higgins, Thoresen, & Barrick, 1999). Oreg (2003) visade med sin studie att individer med låg närvaro av Openness to Experience i sin personlighet visade större motstånd vid förändringar än individer med hög närvaro. Att teammedlemmar är öppna för förändringar kan ses som något man gör och kan bli bättre på, vilket stämmer överens med definitionen av förmåga som används i studien. Därför anser vi att öppenhet för förändring är en förmåga. Grad av öppenhet kan sedan förklaras med hur hög närvaro av personlighetsdraget Openness to Experience en individ har, vilket motiverar oss att kategoriserar förmågan som en mjuk förmåga.

Wiedemann och Schulz (2017) identifierade också ***intraprenörskap*** som en viktig kapabilitet för DevOps team eftersom den möjliggör snabba leveranser av innovativa lösningar vilket är en förutsättning för att uppnå syftet med DevOps. Intraprenörskap innebär att anställda inom organisationer beter sig som entreprenörer (Hisrich, 1990). Precis som entreprenörer tar dessa individer fram nya idéer, implementerar dem och genererar förhoppningsvis värde till verksamheten (Hisrich, 1990). Gartner (1988) menar att entreprenörskap inte kan definieras som en typ av personlighet utan att det snarare är en uppsättning beteenden. Däremot argumenterar Rauch och Frese (2007) för att personlighet avgör en individs benägenhet för att agera på ett visst sätt och påvisar i sin studie att personlighet har betydelse för huruvida en entreprenör kommer att lyckas. Då intraprenörskap är ett agerande hos individer som kan relateras till deras personlighet, kommer vi kategorisera intraprenörskap som en förmåga och mer specifikt en mjuk förmåga.

### *2.4.2 Hårda förmågor*

Wiedemann och Schulz (2017) identifierar i sin studie att DevOps team ska kunna använda diverse verktyg för mjukvara, plattformar och databaser såsom specifika ***verktyg för DevOps***. Detta eftersom användandet av vissa verktyg är en förutsättning för att kunna arbeta med DevOps (Wiedemann & Schulz, 2017). Även Hussain et al. (2017) menar att individer som jobbar med DevOps ska kunna använda diverse verktyg relaterade till DevOps. För att DevOps team ska uppnå kapabiliteten behöver dess medlemmar kunna använda verktygen (Wiedemann & Schulz, 2017) vilket enligt de definitioner vi använder är en förmåga och mer specifikt en hård förmåga. Vidare menar Wiedemann och Schulz (2017) samt Hussain et al. (2017) att DevOps team ska kunna arbeta med kontinuerlig leverans, kontinuerlig driftsättning och kontinuerlig integration. Dessa aktiviteter är en del av DevOps (Shahin et al., 2017; Virmani, 2015) och för att kunna arbeta med dem måste teammedlemmarna använda diverse verktyg (Weber, Nepal, & Zhu, 2016; Wettinger et al., 2014). Av den anledningen kan

verktygen klassas som DevOps verktyg och därför kommer kontinuerliga aktiviteter ingå i förmågan att använda DevOps verktyg i den här studien.

Alla DevOps team ska kunna ta ansvar för en produkt eller tjänst, både när det gäller aktiviteter som ingår i Software Delivery Lifecycle (SDLC) samt företagsrelaterade aktiviteter där planering av budget, tid och resurser ingår (Wiedemann & Wiesche, 2018). För att team ska kunna ta fullt ansvar identifierar Wiedemann och Wiesche (2018) ett antal förmågor teammedlemmar borde ha när de ingår i ett DevOps team.

Teammedlemmar ska kunna använda ***verktyg för backend och frontend programmering*** (Wiedemann & Wiesche, 2018). Alla medlemmar ska kunna lösa eventuella problem som uppstår oavsett om det sker i backend eller frontend. Wiedemann och Wiesche (2018) menar även att DevOps team ska kunna utföra alla aktiviteter som ingår i ***den operativa verksamheten***. Varje teammedlem ska ha tillräcklig teknisk kunskap om produkten eller tjänsten samt problemlösningsförmågor för att kunna analysera och eventuellt lösa problem som uppstår (Wiedemann & Wiesche, 2018). Varje teammedlem i ett DevOps team ska också kunna ***skapa och använda automatiska tester*** (Wiedemann & Wiesche, 2018) med hjälp av diverse verktyg (Callanan & Spillane, 2016). Även Hussain et al. (2017) menar att individer som jobbar med DevOps ska kunna använda verktyg för att utföra ovannämnda aktiviteter. Eftersom teammedlemmar behöver teknisk kunskap samt använda verktyg och tekniker för att utföra de aktiviteter som krävs för att ansvara för en tjänst eller produkt, kommer vi kategorisera ovan nämnda förmågor som hårda förmågor.

Wiedemann och Wiesche (2018) menar också att alla teammedlemmar ska förstå innebörden av ***Service Level Agreements*** (SLAs). Anledningen är att teammedlemmar ska förstå den kommersiella inverkan avtalet har för att kunna arbeta så att produktens kvalitet överensstämmer med avtalet. Holloway (2017) beskriver SLA som ett avtal vilket erbjuds av en leverantör till en annan eller till en kund. I avtalet står det vad leverantören ska leverera samt vilka konsekvenser som följer om leverantören inte uppfyller avtalet. Mer specifikt innehåller SLAs oftast tekniska aspekter som leverantören lovar leverera såsom nätverkstillgänglighet, fördröjning och hastighet (Holloway, 2017). Eftersom SLA innehåller tekniska aspekter som varje medlem i ett DevOps team ska kunna förstå anser vi att denna förmåga är en hård förmåga.

### 2.4.3 Sammanställning av förmågorna

Tabellen nedan är en sammanställning av alla förmågor som det redogjordes för i föregående stycke. Varje förmåga följs av en beskrivning, i vilken litteratur den förekommer samt hur vi har valt att kategorisera förmågan.

**Tabell 1:** Sammanställning av mjuka- och hårda förmågor.

| Förmåga | Beskrivning | Litteratur | Typ av förmåga |
|---|---|---|---|
| Anpassningsbarhet | Teammedlemmar ska kunna ta över arbetsuppgifter från varandra. | (Wiedemann & Schulz, 2017) | Mjuk förmåga |
| Beslutsfattande | Teammedlemmar ska kunna ta beslut själva och stå för konsekvenserna av besluten. | (Wiedemann & Schulz, 2017; Wiedemann & Wiesche, 2018; Shakir, 2009) | Mjuk förmåga |
| Kunskapsdelning | Teammedlemmar ska dela med sig av kunskap inom teamet. | (Wiedemann & Wiesche, 2018; Hemon et al., 2019; Hussain et al., 2017; Melnik & Maurer, 2004; Riege, 2005) | Mjuk förmåga |
| Vilja att lära sig | Teammedlemmar ska vara villiga att lära sig både av varandra och från andra källor. | (Wiedemann & Wiesche, 2018; Hemon et al., 2019; Hussain et al., 2017; Major et al., 2006; Robles, 2012). | Mjuk förmåga |
| Kommunikation | Teammedlemmar ska kunna kommunicera inom teamet. | (Wiedemann & Wiesche, 2018; Hemon et al., 2019; Hussain et al., 2017; Robles, 2012; Jelphs, 2006). | Mjuk förmåga |
| Öppenhet för förändring | Teammedlemmar ska vara öppna för de förändringar som DevOps medför. | (Wiedemann & Schulz, 2017; Hemon et al., 2019; Hussain et al., 2017; Barrick & Mount, 1991; Judge et al., 1999; Oreg, 2003). | Mjuk förmåga |
| Intraprenörskap | Teammedlemmar ska kunna ta fram nya idéer samt implementera dem. | (Wiedemann & Schulz, 2017; Hisrich, 1990; Gartner, 1998; Rauch & Frese, 2007) | Mjuk förmåga |
| Verktyg för DevOps | Teammedlemmar behöver kunna arbeta med verktyg såsom för DevOps. | (Wiedemann & Schulz, 2017; Hussain et al., 2017) | Hård förmåga |
| Verktyg för programmering | Teammedlemmar ska kunna arbeta med | (Wiedemann & Wiesche, 2018; Hussain et al., | Hård förmåga |

| | | | |
|---|---|---|---|
| | verktyg för både backend och frontend programmering. | 2017; Callanan & Spillane, 2016; Hemon et al., 2019) | |
| Den operativa verksamheten | Teammedlemmar ska kunna utföra aktiviteter som ingår i den operativa verksamheten. | (Wiedemann & Wiesche, 2018; Hussain et al., 2017) | Hård förmåga |
| Testning | Teammedlemmar ska kunna skapa och använda automatiska tester. | (Wiedemann & Wiesche, 2018; Hussain et al., 2017) | Hård förmåga |
| Service Level Agreements | Teammedlemmar ska förstå innebörden av Service Level Agreements (SLAs). | (Wiedemann & Wiesche, 2019) | Hård förmåga |

## 2.5 Sammanfattning av litteraturgenomgången

Syftet med studien är att påvisa vilka mjuka- och hårda förmågor hos individen organisationer tar i beaktning vid skapande av DevOps team. DevOps definieras av Erich (2018) som ett samarbete mellan utvecklare och den operativa verksamheten. DevOps team ser olika ut från organisation till organisation (Erich, 2018; Shahin et al., 2017) eftersom varje organisation väljer själv vilken teamstruktur de vill använda (Shahin et al., 2017). Vi kommer att studera de teamstrukturer där nya team har skapats eller nuvarande teams arbetsuppgifter har förändrats.

Organisationer kan ta flera faktorer i beaktning när de väljer ut individer som ska ingå i team, bland annat förmågor, personlighet, intressen, attityder och värderingar hos individer. För studiens syfte kommer vi att fokusera på förmågor. För att besvara frågeställningen redogör vi för vilka förmågor medlemmar i ett DevOps team bör ha som tidigare studier identifierat. Vi har delat in dessa i mjuka- och hårda förmågor. De mjuka förmågor som identifierades var vilja att lära sig, kunskapsdelning, öppenhet för förändring, beslutsfattande, kommunikation och intraprenörskap. De hårda förmågorna inkluderar användning av verktyg för DevOps, verktyg för programmering, kunna utföra aktiviteter för den operativa verksamheten och testning samt förstå innebörden av Service Level Agreements (SLAs).

# 3 Metod

*I detta kapitel kommer vi beskriva hur vi gått tillväga för att samla in data, val av respondenter, genomförandet av intervjuer och en sammanställning av resultat. Vidare kommer vi även att behandla etik och kvalitet med hänsyn till val av metod.*

## 3.1 Val av metod

Vi har valt en kvalitativ metod som tillvägagångssätt för vår intervjustudie. Genom att använda en kvalitativ metod menar Jacobsen (2002) att man är mer mottaglig för ny information. Vidare säger Jacobsen (2002) att en kvalitativ ansats är att föredra när man vill skapa en större förståelse för ett aktuellt område och när man vill undersöka teorier. Det var svårt för oss att veta huruvida organisationer tar hänsyn till ovan nämnda faktorer eller inte vid skapande av DevOps team. Därav blev det naturligt för oss att använda denna metod för att kunna besvara vår frågeställning genom att få en djupare förståelse och en mer nyanserad bild över respondenternas definition och erfarenheter gällande vikten av förmågor vid skapande av DevOps team.

Litteraturgenomgången är baserad på artiklar och journaler inom DevOps, team, förmågor, och sammansättning av team. Sökningen efter litteratur skedde främst genom Google Scholar, AIS eLibrary, LUBsearch och böcker relevanta för att kunna besvara vår frågeställning. För att säkerställa kvaliteten och tillförlitligheten har vi så långt som möjligt valt artiklar och journaler som är peer reviewed. Bryman (2016) förklarar peer review som ett sätt att neka artiklar som inte håller den kvalitet som krävs för att en artikel eller journal skall bli publicerad.

## 3.2 Urval av respondenter

För att kunna utföra vår studie valdes respondenter ut som arbetar i organisationer som jobbar med någon form av mjukvaruutveckling och har implementerat DevOps. För att kunna besvara vår frågeställning var det även viktigt för oss att säkerställa att respondenterna hade erfarenhet av sammansättning av team men även att de arbetar teambaserat med DevOps. Respondenterna som intervjuats har erfarenhet av att skapa DevOps team. För att få en bredare insikt och uppfattning valdes intervjupersoner med olika roller och positioner ut.

Respondenterna valdes ut med hjälp av två olika metoder. Den första metoden som användes var ett bekvämlighetsurval vilket innebär att vi kontaktade personer som vi kände sedan tidigare (Jacobsen, 2002). Snöbollsmetoden var den andra metoden som användes och innebär att en respondent som kontaktades gav tips och rekommendationer på andra personer som var relevanta som i sin tur föreslog personer som vi senare kontaktade för att boka en intervju (Jacobsen, 2002). Samtliga respondenter arbetar och är införstådda med vad DevOps innebär och genom detta säkerställdes det att de hade rätt kompetens för vår studie. En intervjuguide

skickades ut via mejl ett par dagar i förväg till de respondenter som önskade det. Detta var något som vi ansåg positivt då de haft tid på sig att läsa igenom och reflektera över frågorna och vara mer förberedda.

I tabellen nedan redovisas vilka företag som deltog i studien och vilken roll respondenterna innehar.

**Tabell 2:** Genomförda intervjuer.

| Namn | Organisation | Roll | Datum och tid | Intervjutyp/ längd | Appendix |
|---|---|---|---|---|---|
| Tobias Sjölin | Boozt Fashion AB | Project Director | 24 april 2019 klockan 10:00 | Personligt möte/60 min | 2 |
| *Andreas Billeqvist* | *sQills* | *Release Manager* | *24 april 2019 Klockan 13:00* | *Telefon/39 min* | 3 |
| Sheldon Keeping | HiQ | Configuration Manager | 29 april 2019 Klockan 10:00 | Personligt möte/60 min | 4 |
| Anonym ”Urban” | Anonym ”IT-huset” | Senior Service Manager | 29 april 2019 Klockan 14:00 | Personligt möte/50 min | 5 |
| Pierre Stehagen | inRiver | DevOps | 7 maj 2019 Klockan 13:00 | Personligt möte/35 min | 6 |

## 3.3 Intervjuer

Vi valde en semi-strukturerad intervjumetod för insamling av empirisk data. Enligt Bryman (2016) innebär denna metod att man ställer förutbestämda frågor utifrån en intervjuguide, där ett bestämt ämne undersöks. Ordningsföljden och formuleringen på frågorna kan variera från intervju till intervju och behöver därmed inte följa guiden. Vid behov kan kompletterande frågor ställas och genom detta få mer utvecklade svar och en djupare förståelse. Vi valde att ställa frågor av en öppen karaktär för att få så utvecklade och detaljerade svar som möjligt utifrån respondenternas perspektiv och erfarenheter av faktorer som bör tas i beaktning vid skapande av ett DevOps team. Jacobsen (2002) menar att fördelen med öppna frågor är att man kan fånga respondentens individuella tolkningar och åsikter av ett givet område.

Ämnet i intervjuguiden baserades på frågeställningen och med bakgrund av det teoretiska ramverket som tillämpas i studien. Intervjuguiden hade följande teman (1) Respondentens erfarenhet, kunskap, intresse och medvetenhet om DevOps, (2) Team, (3) Förmågor hos individen vid skapandet av DevOps team rörande mjuka förmågor och (4) Förmågor hos individen vid skapandet av DevOps team rörande hårda förmågor. Se appendix 1. Anledningen till detta var att få relevanta svar i förhållande till vår frågeställning som sedan kunde analyseras och sammanställas. Intervjuguiden reviderades för att främja kommande intervjuer i de fall där respondenterna framkom med information som inte uppmärksammats innan intervjun och som vi ansåg vara relevant för vår studie.

Jacobsen (2002) säger att det är lättare att genomföra en intervju ansikte mot ansikte när frågorna till större del är av öppen karaktär. Därav valde vi att prioritera besöksintervjuer framför telefonintervjuer.

Jacobsen (2002) menar att tillförlitligheten på svaren påverkas av miljön där intervjun utförs. En obekant miljö kan göra att respondenten svarar på ett annorlunda sätt än vid en naturlig miljö. Av totalt fem intervjuer var fyra intervjuer personliga och utfördes därav antingen på respondenternas arbetsplats eller på en annan plats som de var bekant med. Den femte intervjun hölls via telefon med respondenten som jobbar på annan ort. Vi ansåg inte att kvaliteten på intervjun förändrades beroende på om den utfördes personligen eller via telefon. Respondenten som intervjuades via telefon lämnade lika uttömmande och diskuterande svar som de andra respondenterna. Vi håller med Jacobsen (2002) att det är lättare att genomföra en intervju vid ett personligt möte då det var lättare att skapa en personlig kontakt och avläsa respondentens kroppsspråk och ansiktsuttryck. Vilket bekräftades efter vår telefonintervju. Vi upplevde att det var svårare att skapa samma kontakt som vid ett personligt möte, det var lättare att avbryta respondenten och följdfrågor ställdes inte lika lätt. Vi håller därmed Jacobsen (2002) att det är lättare ett genomföra en personlig intervju.

Intervjuerna inleddes med en snabb sammanfattning om ändamålet med intervjun och en kort introduktion om oss själva (Jacobsen, 2002). Vi bad även om tillåtelse att få spela in intervjun och informerade respondenten om möjligheten att vara anonym i intervjun. En av deltagarna valde att medverka anonymt. Intervjun fortlöpte sedan med mer allmänna frågor om respondentens roll, position inom företaget, hur länge de arbetat med DevOps och deras övergripande syn på DevOps. Slutligen avhandlades frågorna som ligger till bakgrund för att kunna besvara vår frågeställning. Samtliga intervjuer varade mellan 35 minuter till en timme.

## 3.4 Bearbetning av empiri

Efter varje genomförd intervju reflekterade vi över intervjun och lyfte de tankar och funderingar som skapats. Därefter transkriberades intervjuerna och sparades i separata dokument. Genom att göra detta underlättar man arbetet med att finna och markera viktig information och begrepp samt säkerställa att all data från intervjun finns med (Jacobsen, 2002). Därefter skickades transkriberingarna ut till berörda respondenter för att ge dem möjlighet att göra ändringar och reda ut eventuella missförstånd.

Därefter påbörjades analysen av transkriberingarna där vi inledningsvis höll analysen så öppen som möjligt för att inte förbise information som kunde vara viktig för studien (Jacobsen, 2002). Här upptäckte vi att en av respondenterna inte hade erfarenhet av att skapa DevOps team. Därav ansåg vi att intervjun inte kunde användas i studien. En kodning med hjälp av färgmarkeringar ansåg vi inte vara relevant för oss då vårt intervjuunderlag var så pass rättframt. Därav var det enkelt att hitta värdefulla synpunkter och information men även att hitta samband och mönster i data genererad från intervjuerna (Jacobsen, 2002). Analysen baserades på de olika förmågorna som presenterades i det teoretiska ramverket och intervjuguiden men även av den information från respondenterna som vi ansåg viktig för studien.

Som en sista del i analysfasen påbörjades arbetet med att identifiera likheter och olikheter i svaren från respondenterna.

## 3.5 Validitet och reliabilitet

För att säkerställa en god kvalitet på vår studie valde vi enligt Jacobsen (2002) att ta hänsyn till intern och extern validering. Intern validering innebär enligt Jacobsen (2002) att man utvärderar resultatet av empirin så att det är relevant och giltig. Detta uppnåddes genom att transkriberingarna skickades till respektive intervjuperson som därmed fick möjlighet att korrigera eventuella felaktigheter eller missuppfattningar. Den externa giltigheten innebär en generalisering av resultatet och hur pass tillförlitligt det är (Jacobsen, 2002).

Jacobsen (2002) menar att en studies reliabilitet syftar på hur tillförlitlig och trovärdig den insamlade empirin är. Tillförlitligheten i respondenternas svar kan påverkas av sättet som intervjupersonen ställer frågorna och hur denna person framträder. Detta är något som Jacobsen (2002) beskriver som intervjueffekten. För att motverka denna effekt har vi eftersträvat att utföra och uppträda på ett liknande sätt under alla intervjuer. Dock är det svårt att garantera att ingen påverkan skett men genom att ha detta i åtanke har vi gjort vårt bästa för att undvika problematiken. Jacobsen (2002) menar att besöksintervjuer oftare har en högre tillförlitlighet till skillnad mot telefonintervjuer vilket gjorde det naturligt för oss att fokusera på att boka personliga intervjuer så långt som möjligt.

Jacobsen (2002) skiljer även på planlagda och överraskande intervjuer. Den överraskande intervjun är att föredra om man är ute efter spontana och impulsiva åsikter medan den planerade intervju ämnar sig bättre om man vill ha genomtänkta och utvecklade svar. Den planerade intervjun var den som passade oss bäst. Därav bokades intervjuerna in antingen via telefon eller mejl. De respondenter som önskade ta del av intervjufrågorna innan intervjun fick dessa mejlade till sig ett par dagar i förväg. Huruvida detta påverkade reliabiliteten på svaren är svårt för oss att svara på. Dock ser vi ingen märkbar skillnad på svaren från de som mottog frågorna innan intervjun och de som inte gjorde det.

### *3.5.1 Bortfall av respondent*

Under genomförandet av vår andra intervju framkom det att respondenten inte har erfarenhet av att skapa DevOps team. Istället arbetade respektive avdelning inom organisationen i separata team. Därmed har inte ett skapande av DevOps team gjorts utan de har enbart anammat de tekniska delarna som till exempel olika kontinuerliga aktiviteter. Detta trots att vi under arbetet med urvalet av respondenter noggrant informerade varje respondent om vilka förutsättningar som krävdes för att delta i studien. Vi har således valt att utesluta denna intervju för att säkerställa studiens validitet och reliabilitet. Vi valde dock att inkludera transkriberingen från den här intervjun för att påvisa att intervjun genomfördes.

## 3.6 Etik

Etiska aspekter är något man bör ta hänsyn till när man utför en studie för att säkerställa att fenomenet man undersöker sker på ett korrekt sätt. Jacobsen (2002) nämner tre grundläggande krav man bör ta i beaktande, nämligen informerat samtycke, rätt till privatliv och krav på riktig presentation av data.

*Det informerade samtycket* säkerställdes genom att samtliga respondenter frivilligt valde att delta i intervjun efter att ha informerats om ämnet som intervjun ämnade att behandla samt

fått möjligheten att skapa sig en förståelse över avsikten med studien (Jacobsen, 2002). Då syftet med studien har ett organisatoriskt perspektiv och inte ett personligt bedömer vi att frågorna som ställdes var av mindre känslig karaktär (Jacobsen, 2002). Respondenterna informerades även om möjligheten att vara anonym i undersökningen vilket Jacobsen (2002) menar är viktigt då antalet intervjupersoner är få till antalet. Därmed säkerställdes intervjupersonernas *rätt till privatliv*.

Slutligen säkerställdes *krav på riktig presentation av data* genom respondenternas samtycke till att intervjuerna bandades. Detta för att på ett korrekt sätt kunna återge vad som sades under intervjuerna (Jacobsen, 2002). Detta påverkade även valet att skicka transkriberingarna till intervjupersonerna för att ge dem möjlighet att korrigera eventuella fel och missuppfattningar.

## 3.7 Metodreflektion

Bryman (2016) menar att det är svårt att mäta kvalitativa studier då dessa undersökningsmetoder anses handla mer om ord än om siffror. Vidare menar Bryman (2016) att reliabilitet och validitet är mer applicerbart på kvantitativa metoder och inte alls i samma utsträckning för kvalitativa tillvägagångssätt. Istället har två andra kännetecken identifierats för att påvisa tillförlitligheten och trovärdigheten hos denna metod, nämligen trovärdighet och äkthet (Bryman, 2016). Därav har vi valt att ta hänsyn till dessa kriterier under studiens gång för att säkerställa en så hög trovärdighet och äkthet som möjligt.

För det första har vi under hela arbetsprocessen haft ett kritiskt förhållningssätt till litteraturen och medvetet valt artiklar från journaler som är peer reviewed så långt det varit möjligt för att säkerställa relevansen på artiklarna. Detta för att säkra tillförlitligheten ytterligare.

För det andra var antalet intervjuobjekt relativt lågt till antalet i vår studie vilket kan ha en inverkan på trovärdigheten och äktheten. Därav lade vi stor vikt på urvalet av respondenter då vi strävade efter en hög kvalitet på samtliga. Trots detta visades det sig att en utav respondenterna vid bearbetningen av empirin att inte hade erfarenheten av att skapa DevOps team vilket vi tidigare har nämnt. Således valde vi att utesluta denna respondent för att säkerställa studiens äkthet. Det lades även en stor omsorg på intervjuguiden där större delen av frågorna var öppna för att generera så utvecklande och detaljerade svar som möjligt.

För det tredje var valet av en kvalitativ metod det rätta valet för oss då vi önskade skapa en större förståelse för ett aktuellt område men även att vi ville undersöka teorier och jämföra dessa med verkligheten (Jacobsen, 2002). Således anser vi att detta bidrar till en ökad trovärdighet gällande vår studie.

# 4 Resultat

## 4.1 DevOps

### *4.1.1 Definition*

Respondenterna definierar DevOps likt litteraturen på olika sätt. Sheldon definierar DevOps som ett arbetssätt som tillåter snabb återkoppling, mycket automatisering då han menar att automatisering är din vän och att man ständigt förbättras och produktionssätter kod. Tobias definition påminner om Sheldons då han beskriver det som en rad verktyg som hjälper utvecklare att driftsätta kod på ett enklare, säkrare och snabbare sätt. För Urban är DevOps olika saker och han säger att det är svårt att definiera men förklarar DevOps som en möjliggörare för ett agilt arbetssätt. Han menar även att det är ett sätt att säkerställa kvaliteten på det man gör, vilket kräver att man använder verktygen på rätt sätt. Ett agilt arbetssätt tillsammans med DevOps gör det möjligt att kunna göra mer på mindre tid och att leverera snabbare. Pierre definierar DevOps som ett arbetssätt men som på inRiver istället blivit en roll. Vidare säger han att DevOps är länken mellan utvecklare och den operativa verksamheten. DevOps team har därför ansvar för de kontinuerliga aktiviteterna och automatisering generellt. Däremot arbetar inRiver för att skapa tvärfunktionella team.

## 4.2 Team

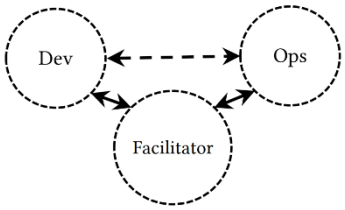
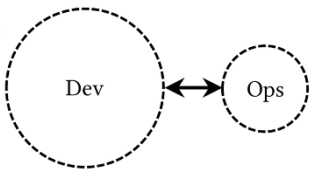
### *4.2.1 Definition*

Tobias på Boozt definierar team mer i form av storlek och menar att antalet medlemmar bör vara mellan tre till fem. Sheldon menar att ett team är en grupp individer som arbetar tillsammans mot ett gemensamt mål. Pierre menar att team bör bestå av minst två personer som har ansvar över ett visst område. Urban beskriver team som de personer som är allokerade för att utveckla och underhålla en viss tjänst.

### *4.2.2 Teamstrukturer*

I kapitel två beskrevs det att vi i den här studien skulle titta på tre typer av teamstrukturer, nämligen B, C och D enligt Figur 1. Vi frågade därav respondenterna vilken teamstruktur av dessa tre de ansåg att de jobbade med. Svaren presenteras i Tabell 3.

**Tabell 3:** Teamstrukturer respondenterna använder (Shahin et al., 2017).

| Teamstruktur | Beskrivning | Respondent |
|---|---|---|
| B   | Finns DevOps team som agerar som förmedlare mellan utvecklingsteam och team från den operativa verksamheten (Shahin et al., 2017). | **Pierre:** Beskriver DevOps team som limmet mellan utvecklare och den operativa verksamheten. Teamet ansvarar för kontinuerliga aktiviteter så att mjukvara kan levereras snabbt till kund. Däremot är målet att skapa tvärfunktionella team med en DevOps roll i varje.<br><br>**Urban:** Placerar sig under den här teamstrukturen men säger i intervjun att varje team har ansvar för en tjänst. |
| C   | Utvecklingsteam ges ansvar för att driftsätta kod med stöd av mindre team från den operativa verksamheten som sköter uppgifter såsom övervakning av system (Shahin et al., 2017). | **Tobias:** På Boozt är det DevOps teamet som ansvarar för de aktiviteter som ingår i den operativa verksamheten. Exempelvis hanterar servers, molnlösningar och skalning. |
| D   | Tvärfunktionella team där både systemutvecklare och anställda från den operativa verksamheten ingår och som ansvarar för en hel tjänst eller produkt (Shahin et al., 2017). | **Sheldon:** HiQ använder sig av tvärfunktionella team som får ansvar för ett projekt från en kund. |

### 4.2.3 Skapa team

Givet de öppna frågor vi ställde om DevOps team gav respondenterna svar som inte enbart handlade om förmågor, vilket vi även påpekade kunde hända i litteraturgenomgången. Tobias berättar att när Boozt rekryterar teammedlemmar externt så kollar de på vilka hårda och mjuka förmågor individen besitter samt att personen ska passa in på företaget. För att avgöra ifall de passar in tittar de på personlighet vilket även är det viktigaste för de när de rekryterar oavsett tjänst. Att passa in blir viktigt eftersom Boozt har skapat en familjekänsla bland de anställda genom att uppmuntra umgänge på fritiden. Detta genom att anordna aktiviteter för dem utanför arbetstid då Tobias har 19 olika nationaliteter på sin avdelning. Väljer de däremot medlemmar som redan är anställda vid Boozt menar Tobias att dessa redan är godkända. Är en anställd intresserad av att jobba med DevOps får den anställde chansen oavsett om personen är kvalificerad på pappret eller inte. I nuläget är alla som jobbar med DevOps anställda på Boozt sedan tidigare. Vidare menar Tobias att på Boozt tycker de det är väldigt positivt att ha individer som är olika i team. Allt från personlighet och nationalitet. Han säger även att det är viktigt att potentiella medlemmar skall passa in socialt i teamet.

På HiQ menar Sheldon att DevOps team sätts ihop när det uppkommer projekt som kräver den typen av team. De försöker då att blanda seniora, juniora och anställda som är där mittemellan. Detta för att de juniora ska lära sig av de seniora och få mer erfarenhet samtidigt som teamet ska leverera ett tillfredsställande resultat. Vidare menar Sheldon att ett viktigt kriterium vid val av teammedlemmar är att potentiella medlemmar ska ha tid över för att ingå i teamet. Om en viss individ verkligen behövs kan man dra ut de ur deras nuvarande projekt, men det görs sällan. Han berättar också att det är väldigt viktigt att tänka på hur väl medlemmarna i ett team kommer att komma överens. Det krävs inte mycket mer än en person för att påverka prestationen i teamet negativt. Sheldon menar att man måste ha en uppfattning om vilka individer som fungerar bra ihop, samtidigt som det är viktigt att känna varje enskild individ. Som Sheldon säger, vissa personer samarbetar väldigt bra med andra, medan andra inte är lika kapabla till det. Det handlar inte så mycket om att medlemmar ska tycka om varandra, snarare att de ska kunna komma överens och arbeta som ett team.

IT-huset anställer cirka 70% konsulter där varje DevOps team oftast består av en fast anställd och resterande del av konsulter. Urban berättar att konsulterna väljs ut främst baserat på hur mycket betalt de tar. Han menar däremot att konsulten självklart måste ha rätt tekniska förmågor, personlighet och passa in, men att priset är avgörande vid valet. Vidare säger han att de team som nu arbetar enligt DevOps har drivit förändringen själva genom att visa intresse.

På både inRiver där Pierre jobbar nu och på CDON som var

hans tidigare arbetsplats, började han jobba med DevOps därför att han visade intresse för att göra det. Därför menar han att det är viktigt att potentiella medlemmar faktiskt ska vilja jobba med DevOps. Vidare säger Pierre att det är viktigt att potentiella teammedlemmar ska passa in med övriga medlemmar för att kunna samarbeta.

## 4.3 Mjuka förmågor

### *4.3.1 Anpassningsbarhet*

Alla respondenter anser att anpassningsbarhet är en viktig förmåga att ta i beaktning när de väljer ut medlemmar till ett DevOps team. Tobias anser den som väldigt viktig för att jobba på Boozt. Han beskriver det som att de inte vill eller är beroende av någon person. Om någon slutar imorgon finns det alltid någon annan som kan ta över. Pierre liksom Sheldon menar att en persons anpassningsförmåga är essentiell för att ingå i ett DevOps team. Vidare menar båda att förmågan är mer eller mindre en förutsättning när det kommer till DevOps. Sheldon menar att anpassningsbarhet är en konstant faktor i DevOps och utan denna förmåga skulle det inte fungera. Att vara beroende av en enda person menar han är riskabelt. Han ser gärna att det finns minst två personer med spetskompetens inom ett visst område i teamet och utöver det ska de även ha övergripande kunskap om resterande uppgifter för att kunna hjälpa till om det behövs. Urban menar att det är viktigt att en person inte skall vara rädd för att hoppa på en sak som man inte är helt komfortabel med eftersom det är ett team. Är någon borta en dag så måste man kanske gå in och göra någonting som man vanligen inte gör. Däremot ser det inte ut så i verkligheten menar han men målet är att nå dit. Han beskriver det som att våga gå utanför sin box. Urban beskriver att man önskar ta in individer som har en T-kompetens. Med det menar han personer som har en bredare kompetens som kan axla andra arbetsuppgifter än bara sina egna.

### *4.3.2 Beslutsfattande*

Beslutsfattande är en förmåga som samtliga respondenter anser är viktig vid ett skapande av DevOps team. Tobias menar att dels måste organisationen ge mandat för att individen skall kunna ta beslut vilket han menar att de gör i sin organisation. Dels är det väldigt viktigt att kunna fatta beslut för det blir svårt att driva verksamheten om inte detta görs. Han förklarar det som om man hela tiden behöver fråga varandra om olika beslut, tappar organisationen drivet framåt. Varje individ har ansvar att driva verksamheten framåt och detta är någonting som måste komma underifrån för att bli förankrat i verksamheten. Både Sheldon och Urban håller med Tobias i hans syn på beslutsfattande. Sheldon lägger ytterligare vikt vid att man måste fatta beslut med hjälp av den information du har tillgänglig där och då. Anledningen till det menar han är den ständiga förändringen av verktyg och teknik, speciellt inom IT-världen där nya lanseras konstant. Sedan får man utvärdera resultatet av beslutet i efterhand huruvida det var bra eller dåligt och vilka åtgärder som eventuellt behöver tas. Urban menar att det man vill uppnå med DevOps är att producera mer per tidsenhet och skall man då behöva be om godkännande för ett visst beslut tappar man finessen med det. Målbilden enligt Urban är autonoma team som kan fatta de flesta besluten själva och göra sina egna prioriteringar utifrån de krav som ställs. Pierre menar att det är viktigt att kunna fatta beslut och sedan argumentera för sin sak.

### *4.3.3 Kunskapsdelning*

Kunskapsdelning är något man tar hänsyn till enligt alla respondenter. Tobias menar att de gärna vill visa vad de gör, vara transparent. På Boozt har man en intern blogg där man kan skriva om något speciellt har inträffat och om man vill dela med sig. Tobias berättar att man skriver vad som hänt, utan namn eftersom det inte är meningen att det skall bli ett “blame

game", hur det gått till och varför det blev som det blev. Vidare säger han att detta är ett bra sätt att få ner kunskapen och för att förhindra att samma fel uppstår igen då man kan gå tillbaka och läsa vad som hände förra gången. Därför tar han den här förmågan i beaktning när han väljer ut teammedlemmar. På IT-huset är situationen annorlunda då företaget använder sig av konsulter i stor utsträckning, vilket Boozt valt att inte göra. Enligt Urban finns det ingen tydlig process över kunskapsdelningen, mer en grundlig dokumentation över en produkt eller tjänst. Det blir en form av kunskapsdelning menar han. Företaget blir sårbart i detta avseende för om en konsult slutar, går en stor del av kunskapen ut genom dörren. Istället hoppas man att omsättningen av konsulter inte blir alltför stor. Därför tar Urban förmågan i beaktning likt Tobias. Sheldon menar att om man är duktig på det man gör måste det finnas en vilja att lära andra.

### *4.3.4 Vilja att lära sig*

Tobias och Sheldon tycker däremot att en individs vilja att lära sig är den viktigaste förmågan av både mjuka och hårda hos en person som skall ingå i ett DevOps team. De anser förmågan nödvändig för att hålla sig uppdaterad, utvecklas, bli bättre och hitta lösningar på problem som kan uppstå. Tobias beskriver det som att en individ i vissa fall kommer att behöva lämna sin kompetensmässiga trygghetszon. Vidare berättar Tobias att på Boozt får medlemmarna i teamet initialt söka svar på lösningar själva och menar att detta kan vara utelämnande i första fasen. Däremot poängterar han att hjälp finns att få men att man måste försöka själv. Urban menar också att den är viktig vid val av teammedlemmar men däremot inte den viktigaste. Pierre anser den som mer önskvärd än viktig.

### *4.3.5 Kommunikation*

Urban anser att fallenheten att kommunicera är den viktigaste av de mjuka förmågorna att ta i beaktning vid val av teammedlemmar. Han menar att kommunikationen spelar en viktig roll när man skall skapa en teamkänsla. Utan den är man inte en del av teamet menar han. Sheldon ser den också som viktig vid val av teammedlemmar och menar att om man inte kan kommunicera och uttrycka sina idéer så blir det svårt att fungera med de andra medlemmarna i teamet. Även Pierre anser att kommunikation är viktigt. Han menar att det är viktigt att kunna kommunicera med andra i teamet om vad man gör och om det händer saker. Speciellt när det sker förändringar i en organisation. Tobias tycker att kommunikation är en förmåga som är viktig att ta i beaktning vid tillsättning av vissa roller.

### *4.3.6 Öppenhet för förändring*

Öppenhet för förändring är en förmåga som samtliga respondenter tar i beaktning hos en person vid val av teammedlemmar. Tobias menar att det är en förutsättning för att jobba på Boozt. Beslut om förändringar kan komma med kort varsel och kräver personer som kan hantera dessa situationer och samtidigt komma med kreativa lösningar. Vidare menar Tobias att de som inte hanterar det lika bra kommer vara fantastiska medarbetare i andra organisationer. Han beskriver att de som tycker att det är jobbigt med förändringar kommer att lämna av sig själva. Urban berättar att det finns personer inom sin organisation som känner sig hotade av de förändringar som en implementering av DevOps medför. Exempelvis kan det krävas att individer behöver ta på sig olika roller och därmed utföra andra arbetsuppgifter, vilket inte var fallet innan DevOps infördes. Därför möter organisationen ett motstånd eller

ovilja från de anställda. Pierre menar att allting förändras vare sig man vill eller inte. Det är därför något som man behöver behärska. Men han säger även att huruvida den tas i beaktning eller inte beror på var företaget är i sin egen process. Sheldon håller med övriga respondenter då DevOps kräver att man hanterar förändring genom att exempelvis ta över arbetsuppgifter av varandra.

### *4.3.7 Intraprenörskap*

Den sista mjuka förmågan är intraprenörskap. Pierre menar att detta är en förmåga som han rekommenderar starkt när det kommer till DevOps då man automatiserar en hel del. Han liksom Sheldon menar att det är optimalt om man har en idé som man sedan kan implementera. Även Urban uppskattar intraprenörskap, att medlemmar i ett DevOps team kan komma med förslag på lösningar, exempelvis nya verktyg som kan användas. Vidare säger han att medlemmarna uppmanas att våga ifrågasätta de arbetssätt som används och komma med förbättringsförslag. Däremot måste man argumentera för sin lösning och få den godkänd. För Tobias är detta något av en förutsättning hos en individ som skall ingå i ett DevOps team. Däremot berättar han att när de rekryterar till hans avdelning letar de efter den förmågan, oavsett om individen ska ingå i ett DevOps team eller inte. Alla anställda ska kunna hitta en lösning på ett problem genom olika typer av forum som exempelvis bloggar och Stack Overflow. De ska kunna läsa mycket och förkovra sig men samtidigt sortera informationen och vara källkritiska, för att sedan implementera en lösning och analysera utfallet.

## 4.4 Hårda förmågor

### *4.4.1 Verktyg för DevOps*

Gällande kunskap om verktyg för DevOps gav intervjudeltagarna olika svar. Pierre och Tobias tycker att det är viktigt att ha en bra grund inom detta. Sheldon och Urban anser inte den vara viktig då det är en förmåga man lär sig under tiden man jobbar. Pierre anser att potentiella teammedlemmar ska kunna använda de stora verktygen såsom Azure från Microsoft, AWS från Amazon och verktyg från Google samt Alibaba. Trots att Tobias tycker det är viktigt menar han att det är viktigare för honom att personen i fråga är villig att testa nya saker än att vara fast vid ett arbetssätt. Urban säger att han tror att många idag vet hur man arbetar med DevOps verktyg. Vidare förklarar han att grundprinciperna är detsamma för alla verktyg och har du jobbat med ett verktyg så har du jobbat med alla. Sheldon anser att förmågan går att lära alla medlemmar när de har gått med i ett DevOps team. Han berättar att man bör ha en, och ifall man har en tillräcklig budget, två DevOps experter som kan lära upp de andra om hur verktygen fungerar.

### *4.4.2 Verktyg för programmering*

Enligt Pierre är det inte viktigt vid val av teammedlemmar att de kan använda verktyg för både backend och frontend utveckling. Han menar att DevOps handlar om att man har ansvar för flödet av kontinuerlig integration och kontinuerlig leverans samt att bygga en miljö så att utvecklarna kan driftsätta koden. Han berättar att det vanligtvis finns någon annan som är just fullstack utvecklare och som tar hand om både frontend och backend. Urban säger att det

första de kollar på när de ska anställa en konsult är deras tekniska förmåga för att sedan gå in på andra förmågor. På Boozt ansvarar DevOps teamet för den operativa verksamheten och det är utvecklarna som behöver denna typ av kompetens vilket sitter i andra team. Sheldon menar att beroende på om ett DevOps team behöver både frontend och backend så blir förmågan viktig vid val av teammedlemmar.

### *4.4.3 Den operativa verksamheten*

Pierre menar att medlemmarna i ett DevOps team ska kunna utföra aktiviteter som ingår i den operativa verksamheten eftersom de ansvarar för de kontinuerliga aktiviteterna. Därför är detta ytterligare en förmåga han tar i beaktning när han väljer ut teammedlemmar. Sheldon har märkt att det inte finns några specifika medlemmar som endast sysslar med den operativa verksamheten i ett team. Mycket av det som förut skulle vara på den operativa verksamheten är nu automatiserat. Däremot menar han att det finns vissa aktiviteter i den operativa verksamheten som de flesta kan utföra. Exempelvis att uppgradera Linux. Tobias säger att på Boozt utför DevOps teamet alla aktiviteter i den operativa verksamheten, exempelvis molnlösningarna och skalning av dessa. Urban anser att eftersom DevOps team ansvarar för en hel tjänst eller produkt så ska teamet kunna produktionssätta och underhålla tjänsten eller produkten. Vidare förklarar han att hur de här arbetsuppgifterna delegeras inte är relevant för honom utan att det viktigaste är att det sker.

### *4.4.4 Testning*

Tobias säger att de lägger ansvaret att testa på utvecklarna. De får genomföra sin egen testning och ska även ha koll på automatisering av tester. Sheldon nämner att det är viktigt att lägga ner mycket tid på testning och att ju tidigare man gör det desto bättre. Däremot berättar han att för honom är det enklare att lära ut hårda förmågor än mjuka. Pierre menar att testning inte är en förmåga han tar hänsyn till vi skapande av DevOps team eftersom DevOps inte ansvarar för det. Urban säger att testning är någonting man kan lära sig under tiden. Han fortsätter med att säga att beroende på vilka andra teammedlemmar som är utvalda tar han hänsyn till förmågan eller inte. Finns det andra som kan testning så blir förmågan inte viktig för vissa teammedlemmar då de kan lära sig när de gått med i DevOps teamet.

### *4.4.5 Service Level Agreement*

Tobias säger att de jobbar med SLA på Boozt men kallar det inte för SLA. Majoriteten av deras system är egenutvecklade och därför har de inte SLA med tredje part. Istället har de interna avtal som liknar SLA där bland annat nedtid specificeras. Urbans företag har ett SLA med sin leverantör av infrastruktur men nämner att SLA inte är någonting som de pratar om varje dag. Pierre säger att det är väldigt viktigt att jobba med SLA och han säger att det är den operativa verksamheten som tar hand om det. Från Sheldon fick vi inget tydligt svar gällande hur de arbetar med SLA.

# 5 Diskussion

## 5.1 DevOps

Våra respondenter ser DevOps som ett sätt att snabbt och kontinuerligt leverera mjukvara och har använt olika teamstrukturer för att anamma DevOps. Respondenterna identifierade sig med en teamstruktur som Shahin et al. (2017) identifierade i sin studie. Syftet med varje teamstruktur är att öka samarbetet mellan utvecklare och den operativa verksamheten. Därmed stämmer även den definition vi använder av Erich (2019) överens med respondenternas användning av DevOps. En enad syn kring DevOps ökar relevansen av respondenternas svar för denna studie.

## 5.2 Team

### *5.2.1 Definition*

Tobias definierar team som en grupp med tre till fem medlemmar. Pierre beskriver team i snarlika drag som Tobias och menar att team bör bestå av minst två personer. Båda respondenternas definition går i linje med Morgan et al. (1986) som menar att team består av minst två personer. Pierre vidareutvecklar sedan och menar att team har ansvar över ett visst område. Hans definition går i linje med Katzenbach och Smith (2005) som menar att team producerar resultat som alla tar ansvar för. Urban specificerar team ytterligare och beskriver ett team som de personer som är allokerade för att utveckla och underhålla en viss tjänst. Detta kan ses som att producera ett resultat vilket stämmer överens med den definition presenterad av Katzenbach och Smith (2005). Sheldon fortsätter och menar att ett team är en grupp individer som arbetar tillsammans mot ett gemensamt mål vilket Katzenbach och Smith (2005), Wheelan (2017) och Morgan et al. (1986) menar att ett team ska göra.

### *5.2.2 Skapa team*

Det framkom av respondenterna att det finns andra faktorer som dem beaktar när dem skapar DevOps team, vilket även framkommit i den litteratur vi presenterade i litteraturgenomgången. Två av respondenterna ansåg att det är viktigt att tänka på hur väl potentiella medlemmar kommer att komma överens. Medlemmar behöver nödvändigtvis inte tycka om varandra utan de behöver enbart kunna arbeta ihop. Respondenternas svar är i enlighet med Wheelan (2017) som också menar att teammedlemmar inte behöver tycka om varandra för att jobba effektivt. De behöver snarare kunna samarbeta. Tobias menar att teammedlemmar ska kunna umgås även utanför jobbet och ha en familjeliknande relation vilket går emot argumentet Wheelan (2017) presenterar. En annan faktor som kom fram var både psykologisk och demografisk mångfald såsom Jackson et al. (1995) beskriver det. Där menar Tobias att han ser det som positivt med båda typerna av mångfald i DevOps team men

att han inte aktivt letar efter det. Därmed kan vi inte fastställa att psykologisk och demografisk mångfald medvetet tas i beaktning.

Det framkom också faktorer som vi inte identifierade i litteraturen. En av dessa är att val av teammedlemmar kan baseras på visat intresse hos anställda. Ytterligare faktorer är priset på konsulter, tillgänglighet på anställda samt en vilja att blanda seniora och juniora anställda. Detta för att de juniora ska lära av de seniora samtidigt som de seniora ser till att teamet producerar ett tillfredsställande resultat.

En faktor vi identifierade i litteraturen men som inte framkom under intervjuerna är nyttjande av informella nätverk såsom Burley (1985) samt Krackhardt och Stern (1988) beskriver det.

## 5.3 Mjuka förmågor

### *5.3.1 Anpassningsbarhet*

Av de svar vi erhållit av respondenterna gällande anpassningsbarhet kan vi tyda att alla anser denna förmåga som betydande vid val av teammedlemmar. Samtliga respondenter menar att det är viktigt att medlemmar i ett team kan ta över uppgifter av varandra. Urban säger att de efterfrågar individer som besitter en t-kompetens. Han beskriver det som en person som har en bredare kompetens utöver sin specialisering och kan hantera olika arbetsuppgifter, inte bara sin egen. Detta ligger i linje med hur Donofrio et al. (2010) beskriver en t-formad profil. Författarna beskriver det som personer som besitter både spetskompetens och en bredare kompetens vilket underlättar för dem att ta över arbetsuppgifter från andra medlemmar i teamet.

### *5.3.2 Beslutsfattande*

Alla fyra respondenter sa att beslutsfattande är en förmåga de letar efter hos individer när de väljer ut medlemmar till ett DevOps team. De motiveringar som lyftes av respondenterna var att IT-branschen ständigt kommer med ny information som teammedlemmar ska kunna använda för att fatta beslut. Fattar de inte beslut så kan inte DevOps team producera de snabba resultat som förväntas av dem och verksamheten drivs inte framåt. Detta ligger i linje med det Wiedemann och Schulz (2017) samt Wiedemann och Wiesche (2018) säger då förändringar på marknaden kan generera ny information som teammedlemmar ska kunna använda för att fatta beslut. Besluten leder förhoppningsvis till produktion av snabba resultat och som driver verksamheten framåt.

### *5.3.3 Kunskapsdelning*

Samtliga respondenter tar hänsyn till kunskapsdelning och anser den som viktig. Wiedemann och Wiesche (2018) menar att kunskapsdelning inom ett DevOps team är betydelsefullt när man skall utöka sina kunskaper och förbättra samarbetet. Då Urban arbetar i en konsultdriven organisation finns risken att en stor del av kunskapen inom teamet går förlorad när konsulternas kontrakt löper ut. Därav blir det viktigt att säkerställa att en kunskapsdelning sker mellan individerna men även att individerna besitter denna förmåga när man skapar DevOps team. Urban menar att de inte finns någon tydlig process för detta utan enbart en

grundlig dokumentation över produkten eller tjänsten. Har man detta upplägg i en organisation så finns det en risk att lärdomen försvinner. Däremot är det ett beslut de valt att ta. På Boozt har man inte den problematiken då man tagit ett aktivt beslut att inte anlita konsulter för att inte förlora värdefull kunskap när någon slutar. Tobias menar även att det är nödvändigt att vara transparent i det man gör för att på så vis dela med sig vad man gör och genom detta öka kunskapsdelning inom teamet men även över andra avdelningar som har tillgång till informationen genom den interna bloggen. Detta ligger i linje med hur Wiedemann och Wiesche (2018) ser på kunskapsdelning. Sheldon menar att om man är duktig på någonting så måste det finnas en vilja att lära andra. Om inte denna vilja finns eller om medlemmarna i teamet inte är mottagliga som Urban upplever det i sin organisation kan kunskapsdelning bli ett problem om organisationer inte vet hur man skall hantera det och kunskapen kan gå förlorad.

### *5.3.4 Vilja att lära sig*

Vidare beskriver Donofrio et al. (2010) att t-formade individer har en stark vilja att lära sig nya saker vilket gör det möjligt att erhålla kunskap både inom sitt eget område men även inom andra. Viljan att lära sig är en förmåga som Tobias och Sheldon anser vara den viktigaste förmågan av både mjuka- och hårda och en nödvändighet för en person som skall ingå i ett DevOps team. Även Urban menar att förmågan är viktig vid skapande av DevOps team. Tobias menar liksom Sheldon att det är viktigt att individerna vill utvecklas och bli bättre. Pierre anser denna förmåga som mer önskvärd än viktig men håller med Tobias och Sheldon att det är viktigt att få en övergripande förbättring och utvecklas. Detta ligger i linje med Wiedemann och Wiesche (2018) som menar att vilja att lära sig är en förmåga som behövs för att kunna utöka sina kunskaper. På Boozt har man fostrat en kultur där man i första hand får leta lösningar på problem på egen hand genom att exempelvis söka information på internet så som Stack Overflow eller bloggar. Denna typ av kultur bidrar oftast till att individerna i teamet hittar problem på lösningarna själva utan stöd från andra inom organisationen. Vi tolkar det som att de har lyckats rekrytera individer med den personlighet som enligt Major et al. (2006) avgör hur benägen en person är att vilja lära sig. Dessa individer behöver inte lika mycket stöd eller uppmuntran från sin miljö för att lära sig i motsats till individer med mindre benägenhet. På organisationen där Urban jobbar möter man ett motstånd när det kommer till att vilja lära sig nya saker vilket kan bero på att implementationen av DevOps gjorts relativt nyligen och att man inte har förankrat detta hos alla inblandade. Det kan även tolkas som att man inte har tagit hänsyn till denna förmåga i samma utsträckning som Tobias och Sheldon har trots att man anser att den är viktig.

### *5.3.5 Kommunikation*

Kommunikation anses vara en förutsättning för kunskapsdelning i ett team (Melnik och Maurer, 2004; Riege, 2005). Vidare identifierar Wiedemann och Wiesche (2018) och Hemon et al. (2019) kommunikation som en förmåga medlemmar i DevOps team bör ha. Sheldon ansåg denna förmåga som den viktigaste av alla då han menar att det blir svårt att samarbeta med andra om man inte kommunicerar och uttrycker sina idéer. Wiedemann och Wiesche (2018) styrker Sheldons uttalande genom att säga att kommunikation bidrar till att skapa goda relationer och ett bättre samarbete inom ett team. Detta bekräftar även Urbans syn på denna förmåga som han menar är viktig när man skall skapa team och teamkänsla. Även Pierre håller med om att kommunikation är viktig för att informera om vad man gör och vad som

händer. Trots att Tobias menar att kommunikation enbart är viktigt för vissa roller så kunde vi baserat på intervjun utröna att Boozt har ett bra samarbete inom teamet, har en fungerande kunskapsdelning och individer som vill utvecklas. Därmed tolkar vi det som Tobias anser kommunikation som viktigt. Detta kan förklaras med hjälp av de kommunikativa förmågorna hos anställda (Hemon et al., 2019). Att Tobias anser att förmågan endast är viktig för vissa roller kan förklaras baserat på den teamstruktur hans avdelning på Boozt använder. När utvecklare ges mer ansvar och den operativa verksamheten mindre, kan vikten av kommunikativa förmågor förändras för anställda på avdelningarna. Däremot kan vi inte avgöra ifall förmågan blir viktigare för utvecklarna eller den operativa verksamheten. Kommunikation är en förmåga respondenterna ser som värdefull hos en individ när det kommer till att skapa DevOps team. Det kan förklaras med att kommunikation kan bidra till att några av de förmågor vi nämnt tidigare även kan utvecklas. Samtliga respondenter tar därför hänsyn till förmågan vid val av medlemmar till ett DevOps team.

### *5.3.6 Öppenhet för förändring*

Öppenhet för förändring är en förmåga som samtliga respondenter menar är grundläggande för DevOps och definitivt bör tas i beaktning när team skapas. Förmågan blir extra viktig då en implementering av DevOps kräver en förändring i samarbetet till det bättre enligt Perera et al. (2017). Däremot menar Pierre att förändringar är något som sker kontinuerligt och därav är detta något som man behöver hantera vare sig man vill eller inte. Han tillägger att huruvida viktig denna förmåga är styrs även av var en organisation är i sin egen process. Openness to Experience, som förklarades i litteraturgenomgången, kan användas för att se hur öppen en person är när det kommer till förändring. Personlighetsdraget kan visa sig i form av nyfikenhet och öppenhet (Barrick och Mount, 1991). Resultatet av studien Oreg (2003) utförde visar att individer med låg närvaro av Openness to Experience i sin personlighet visade större motstånd vid förändringar än individer med hög närvaro. Utgår man från modellen så kan det tolkas att personlighetsdraget inte finns i lika stor utsträckning hos individerna i Urbans organisation. Detta eftersom han beskriver att det finns personer som känner sig hotade av den förändring som DevOps innebär. Däremot kan man anta att dessa personlighetsdrag finns hos individerna på Boozt då Tobias menar att de har den typen av människor som hanterar förändringar på ett bra sätt. Om denna förmåga inte tas i beaktning och inte finns i en större utsträckning i team kan det leda till problem eftersom DevOps medför förändringar. Öppenhet för förändring är en förmåga som respondenterna tydligt menar att de fäster stor vikt vid.

### *5.3.7 Intraprenörskap*

Tobias, Sheldon och Urban tycker att intraprenörskap är en viktig förmåga när de väljer ut medlemmar till DevOps team. Pierre menar att förmågan är viktig eftersom man med DevOps vill automatisera så många manuella aktiviteter som möjligt. Teammedlemmar ska därför kunna identifiera dessa aktiviteter och automatisera dem. Däremot innebär intraprenörskap enligt Hisrich (1990) att individer tar nya idéer, implementerar dem och förhoppningsvis genererar de värde till verksamheten. Att kunna identifiera manuella aktiviteter som ska automatiseras behöver nödvändigtvis inte betyda att man tar fram nya idéer och implementerar dem. Därmed kan vi inte med säkerhet säga att Pierres syn på förmågan stämmer överens med den som används i studien vilket påverkar relevansen av hans svar. Övriga respondenter anser att förmågan är viktig för att team ska kunna skapa nya saker och driva verksamheten framåt. Deras motiveringar går i linje med den Wiedemann och Schulz

(2017) presenterar där de menar att intraprenörskap möjliggör för DevOps team att snabbt leverera mjukvara av hög kvalitet till kund. Tobias letar däremot efter förmågan oavsett vilken tjänst som skall tillsättas. Vi kan därför komma fram till att den här förmågan inte bara tas i beaktning vid val av medlemmar till DevOps utan är en förmåga som alla anställda ska ha på Boozt.

Vi försökte se ifall respondenternas svar kunde förklaras med den teamstruktur respektive företag använder. Alla respondenter var enade kring vikten av förmågorna förutom två. Däremot kunde vi inte använda Pierres svar på grund av olika syn på intraprenörskap. Även Pierres svar angående vilja att lära sig skiljde sig från övriga respondenter. Eftersom företaget där Pierre arbetar använder en annan teamstruktur än övriga respondenter kan hans svar förklaras med hjälp av teamstrukturer. Urban menar att hans arbetsplats använder samma teamstruktur som inRiver men beskriver DevOps team på IT-huset annorlunda under intervjun. Därför kommer vi inte placera Urban och Pierre under samma teamstruktur.

### *5.3.8 Sammanfattning av mjuka förmågor*

Sammanfattningsvis är anpassningsbarhet, öppenhet för förändring, beslutsfattande, kunskapsdelning och kommunikation är fem förmågor som samtliga respondenter uttryckligen menar är viktiga att hänsyn till vid skapandet av DevOps team.

## 5.4 Hårda förmågor

### *5.4.1 Verktyg för DevOps*

Wiedemann och Schulz (2017) identifierade i sin studie att medlemmar i ett DevOps team ska kunna använda diverse verktyg för DevOps eftersom det är en förutsättning för att jobba med det. Förmågan är däremot inte viktig för Urban och Sheldon vid val av teammedlemmar. Båda menar att man lär sig detta medan man jobbar och att det räcker med att man har en expert som lär de andra. I deras fall kan det tolkas som att det är viktigare att en person är villig att lära sig snarare än att de redan kan använda verktygen. Deras svar bekräftar Stevens och Normans (2015) resultat som visade att arbetsgivare anser mjuka förmågor vara viktigare än hårda vid rekrytering då de är svårare att lära sig. Tobias och Pierre menar däremot att förmågan är viktig vid val av teammedlemmar. Både Wiedemann och Schulz (2017) och Hussain et al. (2017) säger att det är viktigt att ett DevOps team ska kunna arbeta med kontinuerliga aktiviteter. För att arbeta med dessa aktiviteter måste medlemmarna i teamet ha kunskap om verktyg för DevOps (Wettinger et al., 2014; Zhu et al., 2016). Baserat på detta menar vi att Tobias och Pierre tar förmågan i beaktning vid skapandet av DevOps team.

Eftersom respondenterna inte alla var eniga kring huruvida förmågan att använda diverse DevOps verktyg ska tas i beaktning eller inte, är det av intresse att se ifall det finns ett samband mellan deras svar och den teamstruktur respektive företag använder. Urban och Sheldon jobbar båda med DevOps team som är ansvariga för en tjänst. Tobias och Pierre har däremot olika teamstruktur men båda anser att DevOps teamet förvaltar arbetssättet och även de ger samma svar. Eftersom respondenterna med samma eller liknande teamstruktur ger samma svar, kan det tolkas som att det finns en koppling mellan teamstruktur och förmågan.

### 5.4.2 Verktyg för programmering

Wiedemann och Wiesche (2018) menar att alla teammedlemmar ska kunna lösa problem som uppstår i både backend och frontend. Detta för att teamet ska kunna ta fullt ansvar för en tjänst eller produkt. Pierre och Tobias jobbar med DevOps team som inte har ansvar för en tjänst eller produkt och sköter därmed inte utveckling av dessa. Respondenterna svarar även att de inte tar denna förmåga i beaktning vid skapande av DevOps team. Deras svar kan därför förklaras med hjälp av den teamstruktur de använder. Sheldon menar att typen av projekt avgör ifall förmågor i verktyg för backend- och frontend behövs. Därmed tas förmågan i beaktning vid behov. Vi fick inget entydigt svar från Urban gällande hans syn på verktyg för programmering. Däremot nämnde han under intervjun att hans företag har många konsulter och att det första de kollar på när de anställer en konsult är deras tekniska kompetens. Huruvida förmågan inkluderas i den kan vi inte avgöra och därför faller Urbans svar bort. Därmed är inte alla respondenter överens om huruvida förmågan beaktas eller inte.

### 5.4.3 Den operativa verksamheten

Förmågan rörande den operativa verksamheten har respondenterna olika uppfattning om. Pierre var den ende som uttryckligen sa att det är en förmåga han tar i beaktning vid val av teammedlemmar. Hans motivering stämmer däremot inte överens med den Wiedemann och Wiesche (2018) presenterar då de menar att den är viktig eftersom DevOps team har ansvar för en hel tjänst eller produkt. Övriga respondenter beskriver att deras DevOps team är ansvariga för att utföra aktiviteterna i den operativa verksamheten men svarar däremot inte på huruvida det är en förmåga de tar i beaktning när de väljer ut teammedlemmar. Sheldon menar att många av aktiviteterna i den operativa verksamheten är automatiserade vilket kan förklara varför vi inte fick ett tydligt svar från honom.

### 5.4.4 Testning

Wiedemann och Wiesche (2018) menar att medlemmar i ett DevOps team ska kunna skapa och använda automatiska tester eftersom de ansvarar för en tjänst eller produkt. Tobias och Pierre säger att testning inte ligger på DevOps teamet utan att det är andra team som tar hand om detta. Dessa två respondenter har olika teamstrukturer så vi kan inte se något samband mellan teamstrukturen och förmågan. Resterande två respondenter säger att kunskaper om automatiska tester är någonting som medlemmarna kan lära sig under tiden de jobbar och att det bara behövs någon som kan det. Detta kan kopplas till Stevens och Norman (2015) som menar på att hårda förmågor är lättare att lära sig än mjuka. Utifrån respondenternas svar så antar vi att detta inte är en förmåga som tas i beaktning vid skapandet av DevOps.

### 5.4.5 Service Level Agreement

SLA syftar på huruvida teammedlemmar förstår innebörden av detta. Holloway (2016) beskriver SLA som tekniska aspekter som leverantören lovar att leverera såsom nätverkstillgänglighet, fördröjning och hastighet. Pierre anser att en förståelse för SLA är väldigt viktigt för potentiella teammedlemmar då det handlar om att driftsätta kod på utsatt tid och att undvika nedtid av systemet, vilket ligger i linje med motiveringen Wiedemann och Wiesche (2018) beskriver. På Boozt arbetar man inte med SLA på det viset utan personal finns alltid tillgänglig vid behov. Urban menar att DevOps team jobbar delvis med det men för honom är inte detta en förmåga att ta hänsyn till vid skapandet av DevOps team. Sheldon

gav inget klart svar på huruvida han ansåg förmågan viktig eller inte. Eftersom endast Pierre anser förmågan vara viktig av de respondenter vi fått svar av, är det av intresse att diskutera möjliga förklaringar till det. Den teamstruktur Pierres arbetsplats använder, skiljer sig från övriga respondenter. Däremot menar han att kännedom om SLA blir viktigt i tvärfunktionella team vilket är målet på inRiver. Då även HiQ använder tvärfunktionella team men Sheldon däremot inte tar förmågan i beaktning kan vi inte se ett samband mellan förmågan och teamstruktur.

### *5.4.6 Sammanfattning av hårda förmågor*

Sammanfattningsvis är det ingen utav de identifierade hårda förmågorna som samtliga respondenter är överens om att man bör ta i beaktning vid skapandet av DevOps team. Delvis överensstämmer vårt resultat med Stevens och Norman (2015) studie som visade att en arbetsgivare lägger en större vikt på mjuka förmågor än hårda vid val av medlemmar till ett DevOps team.

# 6 Slutsats

En implementation av DevOps innebär en stor organisatorisk förändring där man behöver skapa en kultur som möjliggör ett ökat samarbete mellan systemutvecklare och den operativa verksamheten samt att de anställda accepterar detta skifte i arbetssätt. Därav är det av betydande vikt att de individer som skall ingå i ett DevOps team har de förmågor som krävs för att uppnå den fulla effekten och värdet av att implementera DevOps.

Syftet med studien är att påvisa vilka mjuka- och hårda förmågor hos individen organisationer tar i beaktning vid skapande av DevOps team och jämföra de med förmågorna vi har identifierat i litteraturen.

Baserat på respondenternas svar kunde vi identifiera fem förmågor hos individen som alla var överens om är viktiga att ta i beaktning när organisationer skapar DevOps team. Alla dessa förmågor är mjuka förmågor. Dessa är anpassningsbarhet, beslutsfattande, öppenhet för förändring, kunskapsdelning och kommunikation. Testning var den enda förmågan alla var överens om att de inte tog i beaktning vid skapande av DevOps team.

Övriga sju förmågor var respondenterna inte eniga kring. Dessa är vilja att lära sig, intraprenörskap, kunskap om verktyg för DevOps, verktyg för programmering, den operativa verksamheten samt SLA. Dock bör det poängteras att några av respondenterna tar dessa i beaktning när de skall välja individer.

## 6.1 Förslag till vidare forskning

De förmågor som respondenterna inte enades om menar vi kan förklaras med hjälp av teamstrukturer. Däremot hade vi inte för avsikt att undersöka och förklara sambanden, vilket därför gör det intressant för vidare forskning.

Vidare identifierade vi andra faktorer som organisationer tar i beaktning när de skapar DevOps team. Bland annat kollar de på vilka som är intresserade av att jobba med DevOps, vilka anställda som är tillgängliga, hur väl personen kan arbeta i ett team, psykologisk och demografisk mångfald, huruvida personen kommer kunna umgås med övriga teammedlemmar på fritiden samt, för de organisationer som använder konsulter, priset på konsulten. Vilka av dessa faktorer, inklusive förmågor, som i praktiken är viktigast vore intressant att studera för vidare forskning.

# Appendix 1 – intervjuguide

**Respondentens erfarenhet, kunskap, intresse och medvetenhet om DevOps och att arbeta med detta.**

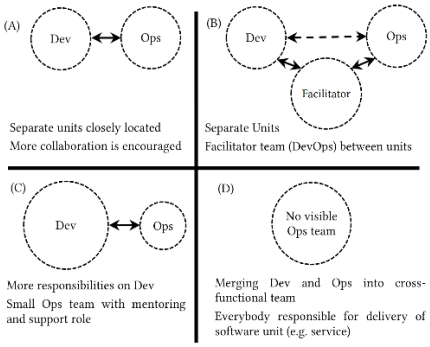
- Vad är din roll/dina huvudsakliga arbetsuppgifter på företaget?
- Hur länge har du arbetat med DevOps?

**DevOps**

- Hur definierar du DevOps?
- Hur ser processen ut när ni skapar DevOps team?

**Team**

- Hur definierar ni team?
- Vilken teamstuktur skulle du säga att ni arbetar efter?




- Vad är det viktigaste att tänka på när man sätter ihop team?
- Hur utvärderar ni vilka individer som skall vara med?
- Vad ställer ni för krav på individerna som skall vara med?
- Varför ställer ni dessa krav?
- Vilka förmågor tittar ni efter när ni skapar DevOps team?

**Förmågor hos individen vid skapandet av DevOps team – mjuka förmågor.**

Anpassningsbarhet – teammedlemmar ska kunna ta över arbetsuppgifter av varandra.

- Vilken hänsyn tar ni till denna förmåga?
- Hur viktigt är den?
- Varför är den viktig/inte viktig?

Beslutsfattande – teammedlemmar ska kunna ta beslut själva och stå för konsekvenserna av besluten.

- Vilken hänsyn tar ni till denna förmåga?

- Hur viktigt är den?
- Varför är den viktig/inte viktig?

Kunskapsdelning - teammedlemmar ska dela med sig av kunskap inom teamet.

- Vilken hänsyn tar ni till denna förmåga?
- Hur viktigt är den?
- Varför är den viktig/inte viktig?

Vilja att lära sig – teammedlemmar ska vara villiga att lära sig både av varandra och från andra källor.

- Vilken hänsyn tar ni till denna förmåga?
- Hur viktigt är den?
- Varför är den viktig/inte viktig?

Kommunikation - teammedlemmar ska kunna kommunicera inom teamet.

- Vilken hänsyn tar ni till denna förmåga?
- Hur viktigt är den?
- Varför är den viktig/inte viktig?

Öppenhet för förändring - teammedlemmar ska vara öppna för de förändringar som DevOps medför.

- Vilken hänsyn tar ni till denna förmåga?
- Hur viktigt är den?
- Varför är den viktig/inte viktig?

Intraprenörskap – teammedlemmar ska kunna ta fram nya idéer samt implementera dem.

- Vilken hänsyn tar ni till denna förmåga?
- Hur viktigt är den?
- Varför är den viktig/inte viktig?

**Förmågor hos individen vid skapandet av DevOps team – hårda förmågor.**

Verktyg för DevOps – teammedlemmar behöver kunna arbeta med verktyg såsom för DevOps.

- Vilken hänsyn tar ni till denna förmåga?
- Hur viktigt är den?
- Varför är den viktig/inte viktig?

Verktyg för programmering – teammedlemmar ska kunna arbeta med verktyg för både frontend och backend programmering.

- Vilken hänsyn tar ni till denna förmåga?
- Hur viktigt är den?
- Varför är den viktig/inte viktig?

Den operativa verksamheten - teammedlemmar ska kunna utföra aktiviteter som ingår i den operativa verksamheten.

- Vilken hänsyn tar ni till denna förmåga?
- Hur viktigt är den?
- Varför är den viktig/inte viktig?

Testning - teammedlemmar ska kunna skapa och använda automatiska tester.

- Vilken hänsyn tar ni till denna förmåga?
- Hur viktigt är den?
- Varför är den viktig/inte viktig?

SLA – teammedlemmar ska förstå innebörden av Service Level Agreements (SLAs).

- Vilken hänsyn tar ni till denna förmåga?
- Hur viktigt är den?
- Varför är den viktig/inte viktig?

Slutligen, är det någon annan förmåga som du anser är viktig vid skapandet av ett DevOps team eller något annat som du vill trycka extra på?

# Appendix 2 – intervju 1

**Organisation**: Boozt Fashion AB

**Intervjuperson**: Tobias Sjölin

**Yrkesroll**: Project Director

**Tid och plats:** 10:00 – 11:00, 24 april 2019, personligt möte, Malmö.

**Vad är din roll, dina huvudsakliga uppgifter på företaget?**
Mitt namn är Tobias Sjölin och jag är då Project Director på Boozt.com och jag sitter på det som heter platform team. Mitt ansvarsområde, nu har jag varit här i 8,5 år, så enades de lite under tiden och jag började en gång i tiden med back-end system ansvar, finansystem och lagersystem och sen blev det mer projektansvar för alla systemen så just nu kan man säga att jag sitter med att försöka få alla teamsen och alla utvecklare att fungera så bra som möjligt i en enda sån stor harmoni. Så jag får in och pilla på alla områden jag vill vilket är jätteroligt.

**Vad är din definition av förmåga?**
En verktygslåda med olika verktyg som man kan använda i sitt arbete.

**Vad är din definition av DevOps?**
Det är en rad verktyg för att hjälpa utvecklare produktionssätta kod på ett enkelt, säkert och snabbt sätt.

**Hur länge har ni jobbat med DevOps?**
Ja, vi har nog alltid jobbat med DevOps men sen är frågan om när vi började kalla det DevOps om det är ett svar på frågan. Så säg att vi från början har vi jobbar med att göra det lätt för utvecklare att sätta saker live och sånt vilket är en del av DevOps och hur det är för någonting. Kan man säga att vi började använda ordet DevOps 5 år sen kanske någonting sånt. Och sen som vi prata om förra gången kallar vi det SRE nu också ibland. Site Reliability Engineer och det är från Google.

**Vad är anledningen till att ni valde att kalla det för det?**
Jo det finns, såhär är det, rent krasst att, det är ibland lättare att få tag i folk om man kallar det SRE istället för DevOps. Så vill man säga att det är någon skillnad på det, inte egentligen, det är samma sak.

**Varför tror du det är enklare att få folk?**
Jo men vissa söker såhär SRE tjänster och då, så ibland är det så att man lägger ut en tjänst så kallar man det DevOps och så då får man inte det här gänget då för de letar efter SRE tjänster. Så man vill liksom, hade jag, jag hade bara kallat det server människa som tar han om servrar och grejer och ser till att det är lätt att utveckla och sånt. Konceptet är ju inte svårare än så. Nu har man ett ord som heter DevOps och så har man SRE och sånt men i grund och botten är detta samma sak som man ska göra, det är inte så stor skillnad egentligen. Sen varierar det på vad man kallar det. Det kan ni också skriva en uppsats om, skillnaden på DevOps och SRE.

**Anser du att DevOps är en roll eller ett arbetssätt?**

Det är lite hönan eller ägget. Vi har börjat med ett arbetssätt som vi inte kallar för DevOps och sen så tillsatte vi en roll, DevOps, för att ta hand om det arbetssättet. Svarar det på frågan? Vi börja arbeta på ett sätt, visst vi hade ett behov då, vi måste ha det här så att det här går fort och snabbt och bra och allting sånt och sen så var det så "aah jaja men det här är ju DevOps ju okej" och då kan vi leta efter en människa som är DevOps, som har hand om arbetssättet.

**Så det var eran process när ni skapa ett DevOps team?**
Ja absolut. Vi hade arbetssättet först och sen ah ja det kan vara bra att ha folk som är ansvariga för det här.

**Du sa innan att ni hade jobbat med det i cirka 5 år. Innan dom 5 åren, hade ni då utvecklarna där och förvaltarna där?**
Nä, det var nästan så att utvecklarna var förvaltarna om du ska jämföra så. Så vi har alltid arbetat, men vi kalla det inte DevOps förens för fem år sen. Så vi har alltid arbetat på det sättet. Utvecklarna i sig vill ju vara så effektiva som möjligt så dom har ju själv såhär "ja men varför kan vi inte bara göra det här? *Något mer servrar*" Jaja men såklart ni ska få göra det prova det. Och sen när man har alla de här små verktygen så bara "Jaja fast det här var rätt så likt DevOps" och sen kan man bara börja kolla finns det mer man kan göra? Det finns dom här bitarna och så kan man ta in den biten.

**Vad är det viktigaste att tänka på när man sätter ihop ett DevOps team?**
Säg såhär, vad är det viktigaste när man anställer en DevOps? Om man bara ska ha en DevOps som blir team då. Om man inte har DevOps och så ja nu ska vi ha det här så i grund och botten så är det som för alla andra vi anställer på platform teamet, att det handlar om personlighet. Personen måste va nyfiken, måste vilja ta initiativ, måste vilja introducera nya saker, måste vara liksom engagerad och driven inom det området så som man vill tillsätta någon på. Så för vi litar mycket på att om man tillsätter en person genom att ha ett nytt team sen om man gör DevOps team att dom själv driver det här för de är om vi hela tiden ska säga ja men nu ska ni göra exakt såhär då blir det, då faller det, det blir väldigt, man kan inte skala det framför allt. Man kan överleva så, man kan göra så i säkert flera år, men man kan inte skala det, man kan inte okej nu ska vi ha 14 DevOps team då blir det jättesvårt om man ska hålla på och micromanagea alla dom där. Så dom måste själv kunna driva det här DevOps och ta reda på egna saker och själv kunna säga "Näe men nu vill vi heta SRE för det är det som e" "ah okej gör det då". Det viktigaste för oss med dom personerna, samma med alla utvecklare och allting sånt, det viktigaste är att man har driv, att man är nyfiken och att man prövar nya saker och att man hela tiden utvecklas. Måste man ha de smartaste människorna någonsin? Nä det måste man inte ha, man måste ha de som liksom (ljudeffekt).

**Har ni DevOps teams och vanliga utvecklar team?**
Just nu så har vi splittat så DevOps ligger utanför utvecklarteamen. Vi hade velat ha att varje utvecklarteam har en DevOps men vi har inte tillräckligt mycket folk för det. Men sen är det vissa i utvecklarteamen som har intresse för DevOps. Så alla våra DevOps som vi har nu är ju från början utvecklare hos oss som vi har gjort om till DevOps istället.

**Har ni en DevOps per team eller är det att för det teamet då är det den?**
Vi hade velat ha det men vi har kanske tre DevOps idag och vi har 75 utvecklare femton team så det blir svårt att sätta ut en på varje. Men vi hade gärna velat ha det. Det hade gett en större förståelse för de problemen som varje team har och kunna sätta sig in i det. Ja men de som jobbar på lagringsystemet de har de här servrarna och de här databaserna och sånt och de har

de här problemen och hur kan vi lösa de problemen? De är ju olika från webshopen som man jobbar med cloudmiljö och sådana saker. Så våra tre DevOps de får hålla reda på det och det och lite överallt.

**Så varje team arbetar på sin kant med sitt?**
Ja. I princip. Så vi har ju att varje team har sitt ansvar för system om man ser helt utvecklarmässigt. Sen har vi knowledge sharing över teamen så alla backenders kanske pratar och alla frontenders och hade man haft DevOps i varje team så hade alla DevOps haft möte om deras teknologi och sånt, så man hade fått den att dela okej vad gör ni på webshopen? Kan vi använda det på lagersystemet eller tvärtom. Så det är viktigt att man delar den kunskapen man kan inte bara sitta med silos.

**Så förmågor och kompetenser som ni tittar på så sa ni driven, nyfiken, vilja ta ansvar?**
Ja initiativ, komma med nya idéer, utveckla och driva, engagera sig, ha ett djupt engagemang i det man gör. Så vi brukar som en bra fråga brukar vara när ni inte får betalt för det ni gör, arbetar, vad gör ni då för någonting? Vad finns det för hobbyprojekt och sånt. Har ni något ni gör hemma och sådana grejer? Då brukar folk prata passionerat om olika saker. "Jag har satt upp en hemsida som visar tv avsnitt" "Ja coolt, berätta mer om det" så får man höra lite olika saker och då brukar man komma åt det som driver människor mer.

**Vi har genom litteraturen så har vi sprungit på förmågor och kompetenser som anses vara viktiga i ett DevOps team. Jag tänker att vi går igenom dom så får du säga hur pass viktigt de är för dig.**

**Beslutsfattande?**
Att man har mandat att göra saker.

**Det är mer om vilken hänsyn ni har till förmågan eller kompetensen.**
Att man kan ta ett beslut? Eller att man har mandat att ta ett beslut?

**Jag tänker att eftersom teamen är självstyrande så måste dom kunna fatta beslut själv utan att springa till dig.**
Rent organisationsmässigt så måste vi ge dem mandat att ta beslut, och det gör vi. Ni får lov att ta beslut. Sen är den andra frågan då, tar de beslut? För det är väldigt viktigt att man tar beslut. Är det viktigt om det är rätt eller fel? Nej. Det viktiga är att ta beslutet.

**Hur viktigt skulle du säga att det är?**
Från 1-5, ska vi rangordna eller? Jag tycker det är viktigt. Det är väldigt viktigt.

**Och varför är det viktigt då?**
För att annars driver man inte verksamheten framåt. Om man hela tiden ska stå och fråga varandra och gå fram och tillbaka mellan olika beslut och så får man inte driv framåt. Och då står det still och det växer mossa och då blir man Sony Eriksson och får lägga ner hela sin verksamhet. (Babbel om att han har sagt värre saker) (12:14). Varje individ har ansvar för att driva verksamheten framåt annars drivs verksamheten framåt. För det kommer inte komma uppifrån. För om det är så att VD:n säger att "Vi måste göra det här" då liksom då dör allting, det blir liksom inte förankrat i verksamheten. Det ska komma underifrån istället.

**Men hur ser ni till på, ni är ju mandat, men hur ser man till på individnivå att dom faktiskt kan ta beslut själva? Även fast de har mandat så ska dom fortfarande kunna göra det.**
Ja exakt. Det är så att de kommer med frågor. "Vad tycker du att vi ska göra här? Tycker du att vi ska göra det här eller det här?" Då säger jag alltid "Jag har ingen aning överhuvudtaget. Jag vet inte det här. Så vad tycker du är bäst? " "Ja men jag tycker detta" "Ja okej men då gör du det" och så gör de det och så blir det fel och så "okej hur tycker du att det gick där? " "Ja men det gick jättedåligt, då borde vi göra det här istället" "Ja gör det istället" Så jag kommer aldrig säga vad man ska göra jag kommer säga så att det övergripande målet för det här är att vi ska ni dit bort någonstans men hur man tar sig dit det får man själv arbeta sig fram till. Det är jobbigt i början, för det kommer att bli mycket fel, men man måste ta den kostnaden för att bygga upp så att det drivs framåt, för annars drivs det inte framåt. Annars kommer de hela tiden vilja gå och fråga och det är jättejobbigt för alla.

**Men det kopplar ni till den här personligheten, nyfiken att testa?**
Ja. Man måste ha kohones och det är inte bara en manlig sak utan det är liksom man måste ha det att liksom, man driver framåt man måste våga. Mod är väldigt viktigt också.

**Mod. Det är också någonting som ni tar hänsyn till och det är jätteviktigt och det är viktigt ur att då vågar man göra saker, fatta beslut, driva.** Det lättaste sättet att få en utskällning här är om man inte gör någonting. Då får man en hårtork (förklaring av hårtork).

**Öppenhet för att lära sig nya saker det kan vi knyta till nyfikenhet tänker jag.**
Ja, jätteviktigt att ta till sig nya saker och prova nya saker. Det hänger ihop för oss ju. Det här med att leta fram nya saker på internet och allt sånt. Litteratur i all ära men det uppdateras inte lika snabbt, så för oss vi är mycket såhär internet, mycket forum, kanske inte forum men det finns stackoverflow, det finns bloggar, ja allting sånt som, det är väldigt bra att läsa och sen ska man inte ta till sig allt. Mycket av det liksom bara, fast, men det är bra att veta det att någon har provat det här och det funkar inte och det är.

**Men pushar ni dom på att göra sådana grejer? Sitta och läsa forum eller utbilda sig själv lite grann, för jag tänker att alla dom här grejerna är ju alla tänker jag såhär men sen när man väl kommer in i jobbet så kanske det inte är så och jag tänker när man sätter ihop teamet har du koll på de här förmågorna och kompetenserna när du nu ska plocka in den här?**
Ja, om jag säger såhär, säg att ni jobbar som DevOps här och så säger ni "Amen vi har problem men den här databasgrejen den här skalar inte som den ska och någonting" och så kommer det till någonting "Hur ska vi göra med det här?" och så säger jag "Jamen har ni letat? Har ni googlat detta?" "UUhh nej det har vi inte" "okej då gör vi det först eller så har ni det" DEt är mycket såhär det här beslutsfattande det faller tillbaka mycket på personerna. Okej ni får själv, hur tar ni reda på den här informationen? Hur kan vi göra här? Och så säger de "Ja men vi har hittat den här ryska bloggen där det står här" "Ah men okej använd inte den kanske, kanske använd något som är mer etablerat än en rysk hacker någonstans där alla lösenord kommer liksom ja". Liksom mer såhär liksom källkritik och allting sånt. Barn lär sig i skolan vilket är jättebra. Jag säger såhär , det är rätt så utelämnande i första fasen sen är det mycket så okej ni får klara er själva men sen är jag såhär sen kommer jag fråga såhär "okej, hur känns det här nu? Känns det som att ni får hjälp eller känns det bara som att det är supersvettigt och ganska stressigt?" Så börjar de "ahnjaah" "Okej men då kan vi hjälpa till då har vi andra liksom så sätter vi ihop så hjälper vi på rätt spår. *Men sen blir det så igen men nu får ni själv titta på det här och se hur det funkar.* Så det är lite, det är inte lätt, mentalt är det

ganska jobbigt ibland, men vi gör så med alla här nästan. Jag pratade med en data scientist igår som vi har sagt såhär "Ja du ska göra en bot som svarar på alla emails så gör den supercool, det blir skitbra" och då sitter han så "Uuuhh ja det är lite komplicerat det här" och då säger jag " Ja, det är det men du är ju expert på detta ju" Så nu har han arbetat med det här i kanske någon månad, nu frågar jag mer "Behöver du mer hjälp? Är det någonting du känner att det här är liksom skakigt, jag vet inte vad jag ska göra och sånt" och då sa han "Nej det här känns ganska bra jag tror jag har en plan för hur jag ska göra" Då säger jag "Jättebra då får du presentera planen nästa gång och ser om vi kan göra det och se om det verkar bra" Svarar det på frågan?

**Ja det svarar på ansvar tänker jag för nu kliver vi in på ansvar för då tar ju han ansvar för den här grejen och så löser han det så vad tar ni för hänsyn till ansvar?**
Ja jätteviktigt. Sa jag det förra gången, den regeln vi har, om något fungerar som det ska, säg om ni bygger upp DevOps och sånt och så blir allting jättebra, då får ni all cred för det. Så då kommer ni "Ja vi har det här (snabbt mummel hör ej vad han säger, 18.25 ish)." Om det går fel så är det mitt ansvar. Så om någon säger såhär "Det här blev ju jättedåligt" så säger han "Det var Tobias som sa det" så går dom och skäller på mig istället, det är mitt jobb. (Vad säger han ens här?? 18.44) Det är ett utbyte(?) att jag vill få reda på vad som gick fel innan någon kommer och skäller på mig för då kan jag säga "Jaja jag vet om det, det är lugnt, vi har koll på det, det blev lite knasigt och så och vi håller på och fixa och såna grejer". Så det är så vi ser på ansvar. Alltså man har alla möjligheter till att göra hur bra som helst om det går åt skogen då tar jag allt ansvar för det, för det är mitt ansvar, till sist.

**Anpassningsbarhet?**
Också jätteviktigt.

**På vilket vis?**
Amen säg såhär, säg att vi har standardservrar så finska servrar och så säger vi om två veckor ska vi byta allting till cloud, gör en plan för det, fixa det. Då kan man inte säga "Ja fast nu var det såhär att vi har ett år som vi ska liksom" Ja fast det är skitsamma. Det är inte det som gäller. Utan det gäller att börja ändra sig, det gäller att ha kreativa lösningar. Jätteviktigt. Det är inte lätt att vara här. Nä, eller jo. Kanske svårt till en början tänker jag. Det passar bra för vissa människor. Vi har en viss typ av människor som passar otroligt bra som är helt naturligt bra såhär ja det blev kanonbra och sånt. Sen är det vissa som inte passar alls för det och dom kommer vara fantastiska någon annanstans.

**Finns det ändå tendenser hos dom som har varit här länge, visserligen byter ni sätt ofta att arbeta?**
Inte egentligen. Om man ser på själva kärnan hur vi arbetar så är den alltid samma. Men sen det vi ändrar är ju vad vi arbetar med, förstår du vad jag menar? Jag hade inte sett det som en stor ändring att gå från php till java, för arbetssättet är samma. Det hade varit värre om vi så nu ska vi ha vattenfallsprojekt, det hade varit en jättestor sak. Men själva verktygen och sånt det är liksom, kallar vi det DevOps eller SRE det är skitsamma. Vi vet vad syftet är med, vi vet varför vi har det.

**För jag tänkte om det finns en tendens att folk som har jobbat här länge att de inte vill byta sätt att arbeta på för att man är bekväm så att säga.**
Tendensen om man har träffat människan som har jobbat här länge så det man nog kommer slås mest av är att de tycker att alla andra är dumma i huvudet. Det är mycket det vi tränar på för att det, vi ligger så långt i framkant så när man pratar med andra så blir det liksom att okej

fast det här är supersegt ju. Så vi försöker mer okej men ska inte ni ändra lite? Kolla här man kan göra såhär. Så liksom "Ah det här brukar ta sex månader att implementera" och så gör vi det på tre dagar. Så det blir en viss kultur här så att okej vi är bäst och sen är alla andra sämst.

**Okej så ni har inget förändringsmotstånd direkt?**
Nä. Dom kommer lämna av sig själv. På gott och ont. För det är jobbigt med förändring ibland.

**Ja för vissa har svårt för det och vissa är mer go with the flow.**
Precis. Sen har vi mer möjlighet nu när vi är så många så har vi mer möjlighet att ha folk som kanske inte man skakar deras värld lika mycket. Det finns möjlighet att sitta i team som kanske är mer stabilt. "Ja okej ni kan sitta det blir jättebra och sånt". Vi behöver ju också en mindutvecklare och det är svårt då att få de där som vi exakt vill ha utan vi kanske också får börja titta lite större och se hur det ser ut.

**Continuous aktiviteter, då utgår vi från continuous delivery, deployment och integration. Hur viktigt är det? Eller vilken hänsyn tar ni till dom här förmågorna och kompetensen?**
Att man har det som tankesätt eller att man?

**Att man anammar det skulle jag säga.**
Ja. Jag ser det nog inte som en egenskap hos en människa utan det är

**Nä asså nu tittat vi ju också på, mestadels utav frågorna handlar om förmågor och kompetenser som man ska tänka på när man skapar team. Men ursprungs egentligen frågeställningen vilka faktorer man behöver ha i beaktande och då är det ju inte bara förmågor och kompetenser. Nu har vi lagt väldigt mycket på och det är det som vi kanske har gått upp för och sett att det är faktiskt inte bara det utan nu går vi faktorer så det är betydligt mycket mer. Som personligheter. Så det här blir väl mer en faktor.**
Jag kan svara på ett sätt så får vi se om det blir bra. Om jag sa så här, om frågan är såhär om continuous integration och continuous om hur viktigt det är då hade det varit noll viktigt. Mer såhär om vi imorgon kallar det för dynamic development då är det viktiga att folk kan förändra sig mentalt. Alltså vi kör inte det här längre, nu ska vi köra det här dynamic development vi hittar det på en blogg från Sydkorea det verkar skitcoolt. Vi gör det här istället för alla continuous grejerna. Då är det viktigare att folk kan förändras och tänka på ett nytt sätt, än att de har ett visst, anpassningsbarhet**,** ja! Sen har vi continuous aktiviteter? Ja. Överallt. Jätte, jätteviktigt att ha det men det är ett verktyg för att kunna göra bra utvecklingsmiljö.

**Så det är snarare att dom ska kunna använda de nya verktygen?**
Ja. Sen vad det är för verktyg, det är skitsamma. Man måste kunna hitta nya för att continuous integration det kanske inte finns nästa vecka för det är helt ute.

**Så man skulle sätta ihop ett team så är det inte det ni skulle värdera högst?**
Nej. "Ah kan du continuous integration? Ja fast det är inte viktigt. Alltså jag vill ha någon som 2020, 2025 har byggt upp en miljö med framtidssäkring".

**Sen har vi, det här är också en faktor då, kunskap om verktyg och teknologier för DevOps? Den kanske går lite hand i hand med continuous.**

Det är bra att ha en grund i det ju men du kan samtidigt säga att du tar en person. Om du har en person. Säg att du har två att välja på. En som kan allting om DevOps och allt sån och sen en som inte kan om DevOps alls. Och den här personen som kan allt om DevOps vill arbeta på sitt sätt och inget annat sätt överhuvudtaget. Sen har vi den här personen som vill ta reda på allting om DevOps och lära sig nya grejer och allting sånt. Då är vi ju helst tagit den personen än någon som inte vill ändra på hur den arbetar. Men om du har en DevOps som kan alla dom här sakerna och vill liksom och säger det här. "Ja jag tittar gärna på nya grejer och sånt och jag har följt den här bloggen och sånt och det verkar jättebra och jag vill gärna prova det i en ny miljö" det är supercoolt ju. Men om man säger så här "Ah jag har jobbat på IKEA i fem år och vi gjorde DevOps och vi kommer göra samma sak hos er." Inte riktigt lika roligt. Sen kan det vara bra också om man får in kunskap och bygger upp den från början. Ja det här var bra för du använde det här på Ikea och det här är en bra grund vi kan arbeta på, men man kan inte stanna där. Allt kommer inte fungera från ett företag till ett annat, man måste kunna anpassa dom bitarna på något sätt.

**Förvaltning hade ni inte riktigt..**
Alltså ta hand om gamla saker?

**Ja men det här är nog utvecklarna, förvaltningen, men ni hade väl inte riktigt någon förvaltningsdel utan utvecklarna**
Men det är, är det DevOps? (Han pratar väldigt lågt här så har lite svårt att höra vad han säger) Förvaltarna? Dom som tar hand om? Definiera förvaltarna.

**I det fallet är det driften av era system så att säga.**
Som vi har det idag så är det samma som DevOps. Dom som är DevOps dom tar också hand om servrarna och allt sånt. Det är inte alla som gör så här. Vissa gör så här att de har utvecklare och så har du DevOps och sen har du de som har hand om servrarna, så är DevOps där emellan. Ibland. Så du har de med servrarna och hur vi hanterar dom och så har vi DevOps som är dom som tar det som är nytt och lägger de på servrarna. Allt det här det är en hög för oss i DevOps. Så dom har hand om allt det här med cloud och skalning och sånt och ser till så att allting funkar hur bra som helst.

**Testning, vad tar ni för hänsyn till det? Har ni nån eller är den automatiserad helt?**
Ja, ingen testavdelning överhuvudtaget. Inga qrs. Det är absolut förbjudet. Det är en strategi, det är ett direkt beslut att vi inte har det på grund av; vi anser att om man har en testavdelning så blir utvecklarna lata. För det man gör i sin utveckling är att man kodar sin grej och sånt och sen så bara släpper man det till testavdelningen, "Ja nu får dom hitta buggar liksom". Och så kommer testarna bli förbannade på utvecklarna, utvecklarna blir förbannade på testarna, för det blir ett sånt blame game där man sitter och säger "Dom gjorde inte rätt och de testa inte rätt, ja men jag har gjort såhär och sånt". Så vi har tagit bort allt det så alla utvecklare ansvarar för sin egen testning. Så de gör sin kod, dom testar sin kod och ser till så att det fungerar. Sen skickar vi ut det live. Funkar det inte då så blir det en hårtork till, sen så drar man tillbaka det, och sen får de ändra, och sen ut med det igen tills det funkar. Sen då som utvecklare då om man nu har en självbevarelsedrift så ser man till att göra automatiska testningar på det här, man testar det man bör testa och sen ska man göra funktionstester och allt man ska testa, unit testing och allting sånt. Så det bakar man in i sin kod. Om vi säger såhär "Aa men du har ansvar för kundtjänst systemet så du ska se till att det inte går ner och att det är så bra som möjligt" Då bygger man själv in det här, amen då gör vi det här för vi vet att den här grejen pajar alltid när vi gör någonting. Okej men då lägger vi in automatiska tester på den så pajar inte den igen då. Sen så bygger man upp det istället. Så vi lägger över ansvaret där. Finns ett

undantag i detta och det är våra appar. Där har vi en testare som testar innan vi släpper ut. Och det är för att appar kan vi inte dra tillbaka lika fort för när man skickar ut en appuppdatering så skickas den ut och sen så är det svårt att ta tillbaka den. Vi kan inte patcha den och så. Med webben är det jätteenkelt för oss är det jätteenkelt att lägga ut och dra tillbaka igen. Det är något som är väldigt bra för oss. Det är en tacksam miljö för oss att arbeta att vi har mycket webbutveckling. För det är lättare att dra tillbaka grejer om det inte fungerar.

**Agilt arbetssätt?**
Ja. Men jag tycker inte om det. Agilt är så uttjatat nu. Det är så "aa arbeta mer agile" Ja. För att om man inte gör det så är det helt vansinnigt. Och sen är det så "ja arbetar ni med scrum?" "Nä vi arbetar inte med scrum". "Arbetar ni med Kanban? (31:08)" "Ja om du måste sätta en etikett på det." Men jag tycker det är samma sak som med DevOps man tar bitar som passar in i verksamheten och får det till att fungera. Så om man ska säga någonting så har vi typ ett Kanban. När man har ledig tid så tar man en ticket och så arbetar man med den och sen nästa. Det är som ett flöde. Vi har inte sprintar med releaser. Vi har inte retrespective och sånt du vet när man sitter ner och utan det gör vi hela tiden. För är det någon som inte tycker om något sätt vi arbetar på så säger de till direkt. "Så jag tycker vi borde göra det här" "Jättebra, gå och gör det då. Ändra på det". Så ja. Det är jättebra. Svar på den frågan om man inte arbetar agilt så är man helt ute och snurrar.

**Därav viktigt?**
Ja viktigt

**Service Level Agreement, jobbar ni med det?**
Ja, men vi kallar det för något annat. Men jag kommer inte ihåg vad det heter nu. Jag kommer på det snart. Såhär är det, att vi har inte SLA och då har man med alla third parties egentligen. Om vi har ett system som vi har köpt från någon annan så finns ett SLA med i det så upptider och grejer och hur vi får support och sådana saker. Vi har ju gjort våra system internt ju, så i teorin så borde vi ha SLA med marknadsavdelningen och sånt. Så att dom vet okej, vad är supporttider och sånt. Nu har vi inte internt. Vi arbetar 24/7. Så vi finns alltid tillgängliga om det är någonting. Sen är det inte uppbyggt att man har övertid och sånt. Vi använder slack som är ett chattprogram och där finns olika kanaler och där kan man skicka ut om det är något som är akut fel. Vi hade en sak igår som inte funka i appen. Så skicka vår HR chef ut det "Det här funkar inte för mig jag kan inte checka ut". Och direkt hade vi sju personer som började kolla på detta. "Ja men vi fixar det, det borde vara det här". Så det är mer så vi arbetar.

**Men jag tänker såhär, dom sju är dom från samma team då?**
Nej. De är helt olika. Det var två från icom, två från oss och lite blandat.

**Men kan inte det blir lite rörigt? Då är alla inne och pillar.**
Ja fast det var mer såhär "Kan ni inte pröva starta om och sånt, finns det någon uppdatering?" och när hon väl hade startat om appen så började det fungera igen. Så vi kunde se att det var bara för henne. Men om det hade varit ett större problem hade någon fått gå in och koda och ändra. Men det är rätt så kontrollerat att göra kodändringar, code reviews och skicka ut det och deploya det och sånt. Så vi går ner i vanliga flödet igen. Jag har inte riktigt svarat på frågan.

**Nä tanken med SLA är att man ska vara medveten om konsekvensen av SLA och att alla ska vara medvetna om att okej, det här är någonting vi måste lösa och vi måste**

**uppfylla kraven. Det var mer det, att alla är medvetna om att dom behöver fixa det så fort som möjligt.**
Vi har ansvarsområden så om kundtjänstsystemet går ner då är det deras ansvar att fixa det så fort som möjligt. Det finns olika nivåer på det, om det är en liten bugg där en person inte kunde skriva in en grej ja det kanske inte är lika farligt. Men den bedömningen gör ju varje team. Det är upp till dom. Om inte dom fixar en bedömning så går någon av oss mer in och skriker på dom. Dom brukar komma till mig när de inte är nöjda med någonting. "Såhär är det, jag har väntat på det här i tre dagar och det här blir aldrig klart." Då kommer jag och säger "Okej, lyssna här nu, nu fick jag det här, vad är det här för någonting. Vad är det som händer?" "Ja det är det här" "Okej, släpp allting ni har, gör det här nu, det är det här ni ska göra." Men det vill jag inte göra, det ska ju vara självreglerande på det sättet ju. Men ibland får vi säga att det här är viktigt, och det här är inte viktigt. Men det är också en utbildningsfas ju. Vi har någonting annat, det är det jag försöker komma på. Om det är errorbudget. Nej det är det inte. Någonting med budget, jag kommer på det snart. Det är såhär, varje team har en budget för fel. Det är typ ett SLA, man får bara ha ett visst antal procent fel i någonting. Så fort det är någonting som har hänt, en bugg eller någonting sånt så skriver man upp det i sin error budget. Det här blev fel och det påverka så här många kunder och det behöver det här och det här. Sen kan man mäta det och det är inte så att man får avdrag på lönen eller något utan man kan se, okej i det här systemet ser det ut såhär just nu och då måste vi göra vissa insatser, vi kanske måste göra en refactoring så ändrar man lite kod och sådana grejer. Eller så sätter vi upp någonting annat med servrar och sådana bitar. Det arbetar vi mycket med. Det är ingen som har bett oss om det, men det är något som vi själv liksom okej vi måste kunna göra det här bättre på något sätt. Hur mäter vi hur bra vårt system är? För det kan vara svårt ibland. Hur många gånger måste vi göra en revert? Hur många kunder blir påverkade av det här och vad är det för direkt pengabelopp vi tappar på ett fel?
Så har vi också gått igenom SLA med business. Inte för att de frågade, men vi satt ned med de och frågade; hur mycket kan ni acceptera att det här systemet är nere? Så kundtjänst, hur mycket kan ni acceptera att kundtjänstsystemet är nere? Och då säger alla, vi kan aldrig acceptera någon nedtid överhuvudtaget. Det måste alltid vara uppe. Då säger vi "Ja, men, om det nu skulle vara det, så hur mycket har ni? Är det en minut vi pratar om, är det en timme? En dag? En vecka?" Då brukar de säga såhär "En vecka kan vi absolut inte" och det förstår vi ju också. "En dag är också väldigt svårt, en timme är kanske okej att vara nere" och vad har vi då för sätt? Kan vi skicka ut ett mejl eller lägga upp på hemsidan att vi har problem. Då tittar vi på sådana grejer. Det har vi pratat med dem och försökt se hur mycket, så lagersystemet kan vara nere fyra timmar, webbshoppen får aldrig vara nere över huvud taget. Så det finns olika system och olika SLAs egentligen om man ser det så.

**Ledarskap, vad tar ni för hänsyn till det?**
Om vi har dom femton teamen så har vi försökt identifiera en person i varje team som är en teamlead. Och där är nästan dom viktigaste egenskaperna för en teamlead är och vara på något sätt visionär, det här systemet det ska vi ta och bli till det här om sex månader och inte bara sitta. När folk frågar "Vad ska vi göra idag?" "Jag vet inte, vi gör samma sak som vi alltid har gjort liksom." Det blir inte bra. Utan det måste vara någon som kan "Okej vi ska på det här hållet, det är hit vi ska". Det är att vara en blåslampa och driva hela framåt. När teamet sitter och "Nä jag vet inte vad jag ska göra" "Nä men gör det här" och sen en dag senare så är det klart eller hur funkar det? Kan jag hjälpa till med någonting? Det är en sån sak om man ska se på agile standup varje dag är bra att göra. Så man träffas varje dag och går igenom jag gör det här, jag gör det här, jag har problem med detta och sånt. Det är ett bra sätt att driva vidare saker.

**Skulle du säga att det är viktigt?**
Inte för alla.

**Inte för alla?**
Nä. så det var svaret på det.

**Amen till viss del kanske? Teamet skulle inte gå under om det inte fanns?**
Nä precis. Vi är inte beroende av någon person här över huvud taget. Det kanske kan vara ett svar. Jag har ingen person i teamet som om dom slutar imorgon och går på dagen så är det något som pajar. Då tar man en annan och "Du är befordrad till det här, grattis, du har nu ansvar för detta, lär dig".

**Så man kan ta över arbetsuppgifter?**
Ja. Väldigt viktigt. Det är också flexibiliteten i det hela. Men ledarskap, vi letar inte aktivt efter ledarskap hos folk vi tar in. Vi kommer ta in folk och sen kommer det nog växa fram under tiden. Här är en människa som är enormt intresserad av databaser, okej fint då kan vi sätta det och nu har du ansvar för det och driva frågan och sånt. Det ser vi på något sätt som en ledarskap funktion, att man kallar in till mötet, kom här nu ska vi ha möte nu ska vi lära oss allting om databaser, jag har dom här grejerna och sånt. Delar med sig av kunskap.

**Men det är inte sådär jätteviktigt i skapandet av team utan det kommer mer i efterhand?**
Japp. Om man ser på platform teamet till en början då. Där var det ändå viktigt att få in lite personer som kunde bygga upp det som vi har gjort idag. Tar man bara fem utvecklare och bara "Här, go crazy".

**Men var det ledarskapet ni var ute efter då eller var det senioriteten i**
Den första som blev anställd var ju CTO:n, alltså min chef Jesper. Han var först.

**Är han visionär?**
Ja absolut. Sen vad det är kan ju vara svårt ibland. Han är med på styrelsemöten. Då har han koll på kanske nio månader till arton månader framåt. Jag har kanske mer koll på noll till sex månader. Det finns en lång sikt och det finns en kort sikt. Så kan man ställa en fråga till exempel; Vi behöver ett nytt utvecklingsteam, i vilket land ska vi lägga det? Hur får vi mer folk? Men sen kommer han inte säga "Vi ska ha blockchain nu". Han kommer inte heller vara den som säger "Nu borde vi göra cloud." utan det måste komma underifrån. Det måste finnas ett behov av det. Okej vi måste göra cloud för att vi kan inte hantera reservkapaciteten längre. Det ska inte bara vara cloud bara för att.

**Men då är det mer botten upp här då?**
Jaja, absolut.

**Men säg att det kommer en sån lösning är det då den personen/dom personerna som kommer med den lösningen, är det dom som får jobba med den?**
Oftast ja. Men sen kanske ifall någon annan är intresserad. "Det där var jättespännande, det vill jag också göra!" "Javisst, vad jobbar du mer? Ja då får jag ta in någon annan". Vi kan inte bara flytta alla. Om alla jobbar på cloud helt plötsligt då blir det lite jobbigt för de andra.

**Men ni drivs mycket av intresse?**

Ja, sen kan det låta som att alla är "wooo" men vi har inte flyttat runt så mycket folk ofta. Generellt sätt ju mer man arbetar med en sak desto mer engagerad blir man i det. Men vill inte släppa dom bitarna. Men ibland när de kommer tillbaka från föräldraledighet då har vi frågat "Okej, vad vill du göra nu? Nu har du ju ändå haft ledigt" Ja då säger han "Jag vill jättegärna jobba med robotarna på lagret." "Ja okej då kan vi sätta dig i det teamet då".

**Måste man ha några kunskaper kring robotar för att kunna byta dit?**
Egentligen inte.

**Okej, man lär sig där?**
Ja. För då har vi så att den som sitter på lagersystemet då han är superduktig. Han kan allt om det. Så om vi sätter in en ny människa där så kommer han lära upp dom. Sen har ju jag lärt honom allting i min tur. Så min uppgift är att göra nya team leads egentligen, som själv kan göra andra team leads. Så jag har tränat upp folk till att göra som jag har gjort och sen ska de träna upp.

**Så det blir så att den med kunskap blir teamlead som sen för vidare.**
Ja. Sen är det de som sitter med kunskap som inte bör vara teamleads. Det finns de hos oss som kan vara enormt duktig och supersmart men man ska kanske inte vara en teamlead. Det finns ju olika, man kan ta ett arkitekthåll eller ett projektledarhåll. Det finns lite olika sådana. Vi brukar inte sätta två titlar så utan det växer fram naturligt. Är det någon som är mer teknisk intresserad, okej det är mer en arkitektroll. Och så finns det dom som är liksom, folk blir inte förbannade på dig så mycket som de blir på andra, och det är mer en projektledarroll, att man kan hantera.

**Kommunikation, hur viktigt är det?**
Det är viktigt i vissa roller. Det är väldigt, väldigt viktigt. Det är viktigt att vara transparent och tydlig med vad man gör, vara öppen med vad som händer. Vi har inga hemligheter överhuvudtaget. Sen är det svårt att få folk intresserade med vad man gör. Så det är svårt ibland att få marknadsavdelningen intresserad av hur vi har satt upp vissa saker och sånt. Men vi vill gärna visa vad vi gör. Vi har post mortems till exempel när någonting har gått fel. Sen har vi haft en större crash eller bugg så skriver vi, vi har en blogg internt, som vi skriver exakt vad som hände, lessons learned och sådana saker på det. Den brukar vi skicka runt till folk för att läsa. Det är en helt utan namn, det är ingen blame game utan det är exakt såhär tänkte vi, så här gjorde vi och så här och därför blev det fel och så borde vi göra nästa gång istället.

**Är det för att få folk att skriva?**
Ja och få ner kunskapen. För de har lärt sig någonting när de har gjort fel. Och så är det bara, vad var det du gjorde fel? Så kan vi gå igenom och så säger han så "Jag ska göra den här databasändringen nu" Fast.. Det var någonting som hände förra gången vi gjorde det. Så letar man så just det, gör aldrig det här för då kommer allting att gå fel. Okej, ja då gör inte vi det här. Det är att få ner kunskapen på ett bra sätt.

**Vid skapandet utan teamen är det mer kanske?**
Ja, kommunikation. Inte till en början. Bra att ha folk med som vill dela med sig också. Ja. vi har haft vissa utvecklare här som kanske inte vart superkanon, helt okej här men som har gått till CTO positioner i start up som har varit fantastiskt duktiga, för att de är duktiga på kommunikation och sånt. Så man lär sig det här också, vi försöker ha det så öppet som möjligt.

**Om vi kollar lite på individer och sånt nu istället. Hur utvärderar ni vilka individer som ska vara med?**
Så vi har en intervju så vi börjar med att någon skickar in ett CV och personligt brev. Sen så läser man igenom det. Då är det såhär, man ska inte generalisera, men man får många från Indien och Pakistan som ansöker och dom har väldigt långa CVs där de sista tre sidorna är fulla av stämplar och när man tittar på ett sånt CV så ser det ut som att de har gjort allt i hela världen och att de är fantastiska. Sen brukar det inte vara så. De visar lite brist på engagemang på ett område. Man vill hellre ha de där "Ja men jag har gjort det här och varit jätteengagerad i det här". När vi tittar på ett CV, först säger vi såhär, ja men har de rätt teknologier, som en utvecklare så är det symfony network som vi har till php. Har man det så är det jättebra. Om man har arbetat med något sorts php framework så är de också helt okej för då kan man lära sig. För en DevOps är det mycket mer, okej, har man arbetat med cloud lösningar, continuous integration och sånt. Så man läser lite sådär key words. Hur ser det här ut? Så verkar inte den här personen vara en seriemördare så okej då kan vi ta in och prata. Så tar man in på intervju och där under intervjun så har vi en teknisk del och en mer mjuk del. Den tekniska delen då brukar vi ställa lite frågor såhär "Berätta lite mer om det här, hur tänker du på det här, har du tittat på den här tekniken och sånt." Så ser man om en person börjar "oooohhh" eller "Ja det är det här, det här och det här". Man känner av lite grann kunskapsnivån. Sen den mjuka delen är mer sådär när man försöker få personen att öppna upp sig lite mer. Prata mer fritt. Vad har du för hobbys? Där ser man lite grann om det blir en bra match in i teamet, det är viktigt för oss att få in rätt person i teamet. Så att man trivs här och att vi trivs med personen. Sen så gör vi test, vi ber personen att göra ett test. Det finns ett DevOps test, det finns ett back end test och det finns ett frontend test. Så man får ett DevOps test helt enkelt och det är mer för att se, jag kan ta mig igenom DevOps intervjuer och få ett jobb, men jag kan ingenting om det. Därför behöver vi göra ett test för att se att man faktiskt kan de här sakerna man har sagt att man kan på intervjun. Det är lite syftet med det så vi inte är helt ute och seglar och att man bara bullshittar igenom.

**Men om ni tar internt då har ni lite koll sen innan vad den människan är, så då behöver ni göra något test.**
Men då är de lite godkända. Då är det mer, jag prata med den här och han vill göra mer av DevOps. Okej ja då bara slänger vi in han där. Det är ingen bedömning så.

**Passar alla in här då? Eller är det någon som du känner såhär, nja?**
Ja men som DevOps?

**Om du skulle sätta ihop ett sånt team, nu har ju ni ett?**
Allihopa i teamet?

**Ja skulle du kunna sätta dom som DevOps?**
Ja absolut. Nä men säg såhär. Vi har några som är mer projektledare som har gått mer till tidsplaner och sånt, pratar med business och sånt. "Åh vi ska bygga finanssystem och sånt". Hade den personen passat in som DevOps? Nä inte på pappret. Men om den personen hade sagt att jag vill jobba med det här för jag tycker det är superintressant och jag vill gräva mig in i det, då hade jag satt den som DevOps. För jag vet att alla kan det, det handlar bara om fokus.
**Vad ställer ni för krav på de här individerna? Då var det teknisk, DevOps testet, vad innebär DevOps testet?**
Ja det är ett test. Det är en fråga som man får besvara.

Det är inte något komplicerat om man kan DevOps. Det är mer att filtrera ut om man inte har någon aning om vad man gör över huvud taget. Det är mer såhär, okej det som php, back end och sånt. På det testet ber vi dom att göra en list page. Bara produkter sådär så kopplar man till en databas. Man får inte lov att använda frameworks och då ser man syntaxer, man ser hur man bygger upp en struktur och sånt. Man måste inte köra men man ska se hur folk tänker i det stora hela. Det är lite fluffigt men man kan se på koden att det här verkar helt okej.

**Hur viktigt är programmering? Både front- och backend?**
Han menar på att antingen är man bra på det ena eller det andra, men att bra programmerare kan båda språken. Han ser dock att man specialist på den ena, men tycker att det är viktigt att man är flexibel och kan finnas i olika projekt. Också att man skall vara nyfiken och inte fastna i tanken.

**Är det några andra faktorer som du tycker, som vi inte har berört. Vi inser att vi går på kompetenser och förmågor. Det finns säkert andra faktorer vid skapandet av DevOps team som vi har förbisett eller inte noterat. Du nämnde personlighet som en.**
Personlighet, det är viktigt att passa in socialt i det teamet vi har här. För då blir allting mycket lättare. Att fråga om hjälp, det känns säkert att arbeta på det här sättet, vilket är viktigt. Google har gjort undersökningar på sina teams, hur ett effektivt team fungerar. För de har olika teams som arbetar olika bra. Nummer ett på den listan, de har fem punkter, men nummer ett, anledningen till att tema fungerar bra det är social säkerhet. Det vill säga att man känner att det är okej att göra fel. Ju mer man känner människor och kan umgås med människor desto lättare är det att göra fel. Sitter man själv och inte umgås med någon över huvud taget och man gör ett fel så blir det en väldigt stor grej för den personen. Det är viktigt rent socialt att passa in i ett team. Sen kan man ta in en person som från början inte passar in men som själv kan sprida det sociala och så. Det behöver inte betyda att det vi gör här inne är en sekt. Det är bra om man tar in någon som man känner att det här är en person som man kan hänga med, även när det inte är jobb. Sen har vi kanske mer av ett familjetänk hos oss. Vi har väldigt många som kommer utomlands ifrån som har flyttat hit. Vi har 19 nationaliteter på våra 35 människor här. Många av dom har inte föräldrar eller vänner här, så vi gör aktiviteter utanför jobbet. “Ja nu ska vi åka gokart här och ja men vi träffas på Espresso House och umgås!” Även folk som har slutat här, jag blev inbjuden igår till att spela innebandy med en som har slutat här för att de saknar folk. Det är viktigt för oss, det har hjälp oss väldigt mycket. Det har hjälpt oss väldigt mycket också och få in dom som är utländska här, så att det inte är svenskt, har vart jättebra för oss. Det är inte så mycket det svenska att man ska sitta ner och man ska vara överens utan det är mer tuta och köra. Det här som ni fråga om, att ta beslut, det är viktigt att göra det. En bra mix av människor är bra. Det är bra att leta efter olika människor. Det är väldigt, väldigt bra.

**Olika på vilket sätt?**
Vilket sätt som helst. Utseende, personlighet, nationalitet, allting sånt.

**Kognitivt tankesätt också?**
Precis. Ja men det skapar en mer dynamisk mix. Om alla är samma så blir det skittråkigt rent ut sagt. Det är samma kvinnligt och manligt att få in en mix av det. Sen anser vi inte att man ska leta efter olika kön eller nationaliteter. Vi har en sustainability rapport som vi har protesterat kraftigt mot vårt team för det stod i den, från Boozt, att vi arbetar på könsfördelningen i bolaget och speciellt platform då arbetar vi på att få in mer kvinnor. Vilket vi absolut inte gör. Vi arbetar för att få in bra folk. Sen vad det råkar vara för någonting det är skitsamma. Det här är en bra människa och sen vad det är för någonting. Det är lite snävt att

se det på det sättet, att det ska vara 50/50 eller att det ska vara en viss procent. Vi har ju fler kvinnor i bolaget än män över huvud taget. Sen har vi dom teamsen som är kanske 80% kvinnor och 20% män. Men sen blir vi utpekade i platform, vilket är jättekonstigt. Vi försöker alltid ta in bra folk som passar bra. Gärna dom som är lite märkliga, som har udda bakgrunder, det är jättekul.

**Finns det några organisatoriska faktorer som man behöver?**
Om man ser på ledarskap från början. Vi har vår CTO och vad han gör, det han har gjort och vad han har gett oss i uppgift att göra under honom det är mycket att skapa en miljö som är uppmuntrat till alla de här egenskaperna. För det är inte alltid lätt att få till det. Så det är något av det största. Det är också svårt arbete att få till så att man inte går in och pillar på detaljer hela tiden. Våga kunna släppa det. Det krävs också att man litar på de människorna som man tar in, att man tar in bra folk om man tar in dåligt folk så blir det dåligt. Man måste lita på folk på det sättet. Så det är mycket sånt. Sen har det varit mycket frustrationer också. Vi har skrikit på varandra en hel del när det inte har fungerat. Så lär man sig något av det. "Hur gjorde vi här nu?" "Det här blev inte jättebra, hur kan vi fixa till så att det blir bättre?". Det är viktigt att bygga upp den här team miljön kan man säga, hur man arbetar. Så att folk tycker det är okej. Folk ska tycka det är okej att komma till jobb, "Ah det första jag gjorde nu var på jobbet var att ta ner hela sidan". Men det är okej, det löser vi, det händer alla. Gör man det fem gånger på raken då är det mer såhär, kan vi sluta med detta? För det börjar bli jätteirriterande för alla.

**Någon annan faktor?**
Nä jag tror det är det. Jag tror mycket på det här med olika människor, det har varit jätteviktigt. Och att ha en säker miljö för alla det är jätteviktigt. Vi har inte börjat med DevOps för att vi ska ha DevOps utan vi har börjat med det av en anledning. Vi hade ett problem vi behövde lösa. Så man inte bara ska, okej DevOps paket 1.0, nu installerar vi det. Utan det var mer "okej kolla här".

# Appendix 3 – intervju 2

**Organisation:** sQills

**Intervjuperson:** Andreas Billqvist

**Yrkesroll:** Release Manager

**Tid och plats:** 13:00 – 13:40, 24 april 2019, telefonintervju

**Jag undrar, får vi spela in det här samtalet?**
Det får ni absolut göra, annars får ni knappa väldigt fort.

**Sen har du möjlighet att vara anonym i uppsatsen, det behöver bestämma nu, utan det kan du fundera på.**
Ja

**Och efter att intervjun är gjord och vi har transkriberat materialet så kommer vi skicka det till dig så att du kan läsa igenom så att du får reda ut om vi missförstått någonting eller nåt.**
Ja, det låter bra.

**Vad bra! Och du fick frågorna på mejlen?**
Jag fick frågorna men har inte hunnit titta så mycket dock men jag hoppas att jag kan ge lite svar i alla fall.

**Det tror jag! Men du, så sparkar vi igång lite.**
Ja

**Vad är din roll eller huvudsakliga arbetsuppgifter på företaget?**
Jag började 2013 som Configuration Manager och har glidit över mer och mer till ren Release Manager för tillfället och överlåter scenbitarna till annat folk.

**Hur länge har ni eller du jobbat med DevOps?**
Ja, det beror lite på hur man ser det men enligt Kriminalvården så har jag väl gjort det sedan jag började 2013. I praktiken så har jag väl försökt implementera DevOps i organisationen sedan dess men det har inte alltid gått så bra eller det har varit ganska problematiskt, en väldigt trögrodd organisation eller myndighet.

**Ja, ok, på vilket vis?**
Ja, det är mycket byråkrati, mycket politik, svårt att göra få igenom förändringar även om man beslutar om förändringar så är det inte säkert att det händer någonting utan det ramlar gärna mellan stolarna.

**Ja, ok så då kan man säga att det är sedan typ 2013 som du har jobbat med DevOps i eller försökt åtminstone.**
Ja, riktig DevOps eller när vi har börjat implementera är bara kanske ett eller ett och ett halvt år tillbaka. Som jag ser det, när vi började få ordning på den här organisationen. När man började jobba som vi vill.

**Vad är din definition på DevOps? Hur skulle du liksom, för när vi har läst litteraturen också där så finns det väldigt mycket olika definitioner på DevOps.**
Ja,

**Om du skulle definiera det, hur skulle det låta?**
Jag ser att det finns fyra huvudpunkter som man bör implementera i organisationer eller hantera just för att kunna jobba enligt DevOps. Det är att hantera configuration management, det vill säga att alla komponenter skall vara källkodshanterade i ett och samma system. För att man skall kunna skapa förutsägbara resultat och man har kontroll över vilken release av mjukvaran som är installerad var. Den scenbiten täcker även in hårdvarubitar och andra mjukvaror som huserar i samma miljö. Det bör man också ha koll på. Sen bör man implementera continuous integration, det är viktigt. Alla delar skall enhetstestas på samma sätt och det gör man lämpligtvis genom automatiska byggen och tester. Sedan har du continuous delivery, det är ju release hanteringen egentligen som också bör automatiseras ända ut till produktion. Det är ju en svår grej att göra oftast i många organisationer. Det krävs oftast ett driftstopp för att man skall kunna leverera till produktion till exempel är att man behöver uppdatera en databas eller någonting sånt där om man gör det manuellt. Så kör vi här till exempel på Kriminalvården. Och sedan är det väldigt viktigt med continuous feedback det vill säga återkopplingen både från byggen, byggserverar och även från produktionsmiljöerna som finns så att man har en kontinuerlig uppföljning av användningen av programvaran eller det man levererar. Och även felstatistik och här är applikationsövervakningen en viktig del man har. Ja, det är väl så jag ser på DevOps.

**Ja, jättebra! Vi har valt att titta lite grann på liksom vilka faktorer man behöver ha i beaktning när man sätter ihop ett DevOps team.**
Ja, jag såg den frågan. Jag är inte riktigt överens om att DevOps är team utan det är ett koncept som skall genomsyra hela organisationen. Så det är inte Pelle, Olle och Kalle som utgör DevOps teamet och sköter allting utan det skall genomsyra hela organisationen och alla skall vara med spåret för att det skall lyckas men det är min åsikt.

**Ja, men det är bra. Den frågan har skapat lite funderingar på dom tidigare intervjuerna vi hade också. För vissa ser det som en, använder DevOps som en roll och vissa ser det som ett arbetssätt att gå efter. Men på något vis, om man tänker, om man skall anamma DevOps vad är det viktigaste en organisation behöver tänka på? När man liksom skall göra, anamma och jobba enligt DevOps?**
Samarbete. Fungerar inte samarbetet mellan både individer och delar av organisationen så är man ju körd.

**Är det en grundsten skulle du säga?**
Absolut!

**Och när du tänker så här avdelningar inom organisationen är det någon speciell?**
Det kan vara Devbiten och eller utvecklingen, driften till exempel, IT-support, alla måste liksom vara med på tåget, det måste finnas tydlig kommunikation där mellan. Och alla skall veta hur man skall gå tillväga.

**Finns det vissa förmågor och kompetensen som man liksom, jag vet inte hur skall jag säga men som är mer viktiga liksom när man väljer att jobba enligt DevOps?**
Ansvar.

**Ansvar.**
Ja, det är jätteviktigt. Särskilt om man tittar på den organisationen jag är nu. Där ansvarstagandet har varit väldigt lågt. Det har funnits en leveransavdelning där jag bland annat ingick där man liksom ”ja nu har vi utvecklat klart här, nu kan vi släppa det” och sen bryr de sig inte mer om det. Och det fungerar ju inte riktigt när man skall jobba enligt DevOps. Alla behöver ta ansvar, känna ansvar.

**Vad blir det för konsekvenser om man inte tar ansvar skulle du säga?**
Det blir en situation som Kriminalvården sitter i idag, alla är beroende av mig och en person till för att kunna leverera kriminalvårdsregistret till produktion och till andra miljöer. Och mitt kontrakt löper ut snart och min kollega är inte heller speciellt sugen på att vara kanske och då sitter dom där i båten och kan inte leverera på ett säkert och bra sätt med hög kvalitet.

**Vad intressant att du sa det för vi hade en intervju tidigare idag som inte använder sig utav konsulter överhuvudtaget, dom väljer istället att anställa av just den biten som du säger att dom blir så sårbara när dom har konsulter för att när dom slutar så dels går dom ut med kunskapen ur dörren dels att de kan sprida med sig kunskapen liksom, som dom har internt liksom.**
Ja, just det. Ja, det är nog en bra eller bra grej tänkte jag säga eller en mix är inte fel om man nu skall ha konsulter och då är ju kunskapsöverlämning väldigt viktigt till fast anställda. Och det är ju det jag sysslar med nu, mycket utbildning.

**Vi har identifierat, ja, du kanske har frågorna framför dig men, vi har ju identifierat genom litteratur och tidigare studier förmågor och kompetenser, nu utgår det från ett skapande av DevOps team men vi får väl utgå från ett anammande i ditt fall då.**
Ja, absolut.

**Ansvar anser du liksom är superduper viktigt, hur ser du på beslutsfattande?**
Det är inte lika viktigt längre om man kommer en bit i DevOps arbeten utan ser mer att det är ett kollektivt ansvar som då är samarbetspräglat, man har en samarbetspräglad kultur där alla känner sig delaktiga och känner ansvar. Och då tar man kollektiva beslut helt enkelt. Alla är med på tåget. Men det kräver som sagt var öppen kommunikation, samarbete och tillit till alla som är med inom organisationen.

**Men tycker du att det hänger ihop någonting med ansvar? För jag tänker att om man kan ansvar för du sa att de är lite beroende utav dig och din kollega att dom kommer och frågar er.**
Ja.

**Kan man dra en parallell då att de inte riktigt är där, där dom känner att de kan fatta besluten själv?**
Så är det, absolut, här ja. Dom är rädda faktiskt. Du har skrivit om mod här också och det har jag också satt som viktigt för det har jag sett här under utbildningen att många i teamen är trädd för att till exempel patcha i produkionen. Vad händer om det går fel när man gör det här under drift. Hur går jag tillväga då**?**

**Hur gör ni för att förebygga det tänker jag?**
Enligt mig så har vi väldigt bra tester i en pre-releasemiljö som vi kallar det och det har fungerat jättebra. Ja, det är väldigt sällan det har gått väldigt snett.

**Vad händer som det går supersnett?**
Ja då får man dra i stoppknappen tänkte jag säga men då är det ju att kunna fatta beslut och kunna känna ansvar och veta vad man skall hitta på och ha kunskap om hela systemet och inte bara den lilla delen som man sitter och utvecklar.

**Nu när ni sen lämnar över liksom nu ifall ni inte förlänger där, kan man, kommer ni typ, vad skall man säga, identifiera nyckelpersoner som kommer fortsätta jobba med**

**implementeringen för att liksom fortsätta det arbetet eller vad tror du kommer att ske?**
Så är nog deras tanke för dom har utsett, nu skall vi se fem, sex, sju personer som vi utbildar på samma sätt här nu som skall kunna ta det jag och en kollega gör idag. För att sprida kompetensen och fler som kan ta ansvar och liksom driva det här framåt.

**Vet du hur man har valt ut dom här personerna?**
Det är nog faktiskt frivilligt. Dom som kände sig manade. Dom som tycker det här är roligt.

**Perfekt. Ytterligare en förmåga är öppenhet att lära sig nya saker. Man kunna dra en parallell med nyfiken skulle man väl kunna säga. Vad tar man för hänsyn till en sådan förmåga eller kompetens? Är det viktigt?**
Jag tycker det är viktigt. Idag inom en sådan organisation, Kriminalvården så har man dålig kompetens just runt dom bitarna som vi jobbar med som till exempel leverans, vilka steg måste man gå igenom för att säkerställa en leverans så att det blir bra kvalitet, vilka olika integrationer finns mot andra system och tredjepartsprodukter som kan påverkas. Så där har man, så det är viktigt tycker jag.

**Vad säger du om anpassningsbarhet?**
Jag tycker det är viktigt allting. Det kommer att ställas nya krav på dom här teamen som finns när man går mer och mer mot DevOps och att man blir själva ansvariga för det som levereras till produktion. Nu är dom ju det indirekt men det hamnar ju ofta i vårt knä först och så får vi försöka lösa det men sen om vi inte kan göra det då hamnar det tillbaka i teamen.

**För det handlar ju lite om det som du var inne på innan, att alla teammedlemmar skall kunna ta över arbetsuppgifter från varandra och i ert fall så blir det väldigt relevant nu liksom ifall ni inte finns kvar så länge till.**
Ja, precis. Uppstår det till exempel ett akut stopp i produktionen så måste ju någon eller några från teamen kunna släppa sina utvecklingsaktiviteter för att istället fokusera på att lösa de här problemen. Så det är ju en anpassning som man måste göra. Mot idag där dom i stort sett bara sitter och utvecklar funktionalitet.

**Så om någonting händer nu till exempel så är det ni som kliver in och löser?**
Ja, och det vet ju IT-supporten om, de går direkt till oss fysiskt. De vet att vi oftast kan lösa det och där måste man hitta andra vägar.

**Det blir lite bekvämlighet där då.**
Ja, precis.

**Mod har vi gått igenom och continuous aktiviteterna var väl de som var grundstenarna för er eller vad du tycker. Hur ser du på kunskap om verktyg och teknologier som rör DevOps?**
Det ser jag inte som så viktigt. Ett verktyg är oftast ganska lätt att lära sig, ett nytt man behöver och det är ju inte själva verktyget i sig som gör att man kan leverera, implementera DevOps i en organisation utan det kan ske på många olika sätt. Saken är att man kan implementera de här fyra hörnstenarna vi pratade om på ett eller annat sätt, sen hur man gör det är upp till varje organisation.

**Förvaltningen, hur delaktig är den hos er?**
På Kriminalvården är förvaltarna väldigt delaktiga, där sitter en enorm verksamhetskompetens. Dom kravställer, dom testar och i teorin tänkte jag säga men eller enligt den leveransprocessen jag har tagit fram en gång i tiden här så måste allt som levereras förankras på ett eller annat sätt i förvaltningen. Annars kommer det inte att bli bra, då kommer det liksom simma omkring saker i produktionen som ingen har ansvar för och som ingen vet

vad man skall göra med om det går fel eller förstör eller någonting sådant. Så allt som levereras till produktion skall vara väl förankrat, något typ av kontrakt med förvaltaren eller mottagare av det man levererar.

**Så vi skulle säga att det är rätt så viktigt att dom också är med och …**
Absolut.

**Du nämnde testning, är det manuell testning eller har ni blandat manuellt och att det någon som utför testerna eller har ni bara att det är förvaltarna som testar?**
Det är alldeles för mycket manuella tester, för min smak på Kriminalvården. Man kör väldigt långa QA perioder, för att testa igenom i stort sett hela applikationen varje gång man gör minsta leverans till produktion vilket är waste of time enligt mig. Men man vill absolut vara säker med hängslen och livremmar så att saker och ting fungerar när det väl hamnar i produktion. Sen har dom ju såklart enhetstester som körs för att hitta de här mest fatala felen som utvecklarna kan göra när de checkar in i källkodssystemet, dom körs automatiskt varje gång du checkar in någon kod.

**Är det här en faktor du anser relevant eller har koll på eller styr på?**
Ja, absolut. Den, skall vi se, branch merge strategi som man har på Kriminalvården är ju inte optimal för DevOps. För det är ju först när man kommer i QA som kodbasen blir samman mergad med och man verkligen kan testa alla integrationer och det leder ganska ofta till snabba rättningar ganska nära leveransen och det är ju absolut inte optimalt för att leverera en programvara med hög kvalitet. Sådär behöver man nog tänka om lite i processen hur man arbetar helt enkelt. Där har man en stor grej att göra organisatoriskt faktiskt, man behöver ändra på det.

**Skulle ni säga att ni arbetar efter ett agilt arbetssätt?**
Nej, det tycker jag inte. Dom tycker det men jag tycker inte det, mer vattenfall. Det är långa projekt, mycket utvecklingstimmar läggs ner innan det hamnar i produktion. Det blir stora kodmängder och merga och dom verkar ha svårt att avgränsa sina releaser vilket gör att det oftast tar väldigt lång tid för dom att utveckla det.

**Men blir det inte svårt att anamma DevOps då tänker jag?**
Jo, det är det. Dom får nog börja tänka om lite och avgränsa sig mer, absolut. Vi har ju inte mer än kanske fem leveranser på ett år vilket är extremt lite och då är det ofta mycket kod som sätts i produktion. Det gör ju att man liksom inte får någon återkoppling heller från så kallade affärssidan. Där behövs ett omtänk och att dela upp, modellisera applikationen mycket hårdare än vad den är idag.

**Hur förbättrar ni samarbetet då tänker jag?**
Mellan?

**Genom hela organisationen tänker jag liksom, ja men i detta fallet är det väl utvecklare och förvaltning och dom som liksom, för samarbetet verkar kanske gå lite trögt också.**
Ja, det gör det ju ofta och det gör ju att man låser upp förvaltarna under väldigt lång period så man deltar mycket i utvecklingsarbetet med kravställning. Så där krävs det ju, dels att man avgränsar delar i applikationen mycket, mycket mer med hårda boundries så att du kan leverera bara små delar som kan tillföra kundnytta. Idag så måste man rent tekniskt leverera ganska stora delar varje gång för att man har en legacy som inte är helt optimal, gamma kod, gammal teknik så där tittat man också på hur man kan ändra tekniken för att lättare kunna leverera till produktion.

**Bra, om vi kollar på SLA, Service Level Agreement?**
Där har jag faktiskt ingenting att tillägga. Det är ingenting som jag hanterar.

**Nej, då hoppar vi över den. Hur viktigt är, om man säger att det finns ett tydligt ledarskap eller en coach? Tar ni hänsyn till det i det här fallet när ni skall anamma DevOps? Hur viktigt är det att det finns en sådan person som kan liksom ha ett övergripande ansvar?**
Det kommer att finnas en som är ansvarig just för DevOps biten, en person som skall kunna coacha in både nya medarbetare och sådana. Få med dem på tåget helt enkelt, DevOps konceptet.

**Är det viktigt att en sådan person finns?**
Ja, jag tror det, iallafall i början innan man får snurr på allting, absolut. Det kan vara väldigt viktigt.

**Har ni någon form utav coach eller någonting inom teamen som det ser ut nu liksom eller är det ni som blir dom personerna?**
Ja, det blir vi nu fram till vårt kontrakt går ut samtidigt som vi då utbildar den här tjejen som skall kunna ta det här ansvaret. Vi har ju även, av någon konstig anledning sköter vi utbildningen av nya konsulter som kommer in till Kriminalvården. Det är inte en anställd som gör det vilket säger en del om organisationen kanske.

**Ja, ni är viktiga för dem i alla fall.**
Lite grann.

**Ja, om man tittar på organisatoriska faktorer, vilka organisatoriska faktorer skulle du säga är viktiga för ett anammande av DevOps?**
Oj, bra fråga.

**Miljön på företaget till exempel typ eller, stämningen i utvecklingsteam respektive förvaltning?**
Ja precis, nämen det är väl prestigelöshet tycker jag är väldigt viktigt. Och att man, just det med det kollektiva ansvaret att man känner det för helheten inte bara för den lilla delen jag sitter utvecklar just nu.

**Känns det som att Kriminalvården har fått med sig personalen i ett anammande av DevOps? Jag får en känsla av att de inte riktigt är med på tåget.**
Nej, organisationen är inte mogen än, det kan jag säga.

**Men hur har den här implementationen initierats från början liksom? Är det ni som har gjort det eller?**
Det är vårt intresse av att leverera hög kvalitet i det vi gör. Så det är helt och hållet vår förtjänst om man säger så. Dom pratar om DevOps men mycket mer än så blir det inte när det väl kommer till kritan så finns det, vad skall jag säga, stuprörsdelar inom organisationen och man håller väldigt hårt på sitt och vill absolut inte ta något ansvar för det någon annan gör. Den mentaliteten finns absolut. En rädsla tror jag, just om jag pratade om att det går fel någon gång.

**Hur långt skulle du säga att ni kommit i en implementation av DevOps?**
Rent tekniskt har vi kommit en bra bit, vi har automatiserat alla byggen, vi har ingen manuell konfiguration någonstans när du sätter upp en ny miljö, allting körs virtualiserat både på server- och klientsidan. Vi har gått från två källkodssystem och två leveranssystem till endast ett system som hanterar både källkod, byggen och leverans. Det finns automatiska tester vid varje incheckning och i källkodssystemet.

**Så den springande punkten här är organisationen i sig och individerna liksom?**
Ja, precis och där springer vi in i väggen stup i kvarten. Det låter bra på möten och alla nickar

och sådär men sen när det väl skall försöka implementeras i organisationen så händer det inte mycket tyvärr.

**Är det någonting annat som du tycker är viktigt om man utgår från det här perspektivet liksom med när man skall anamma DevOps eller om det är andra förmågor, kompetenser eller faktorer som är viktiga att tänka på förutom dom du nämnde från början?**
Man skall absolut inte försöka implementera hela DevOps på en gång, allt på en gång. Då lär det bli pannkaka av det utan kör små steg, väl förankrade steg i organisationen så att dom verkligen känner att de är med på tåget.

**Och vilken är den viktigaste delen att få med sig först? Är det liksom att få med sig ledningen eller?**
Nja, jag vet inte det. Det finns väldigt många obstinata personer inom organisationer som sätter sig på tvären vilket kan förstöra väldigt mycket så mycket så man nog. Ja… Jag tror man skall börja nerifrån faktiskt. Från teamen och kommunikationen med drift, dom bitarna eftersom drift oftast vill ha en stabil miljö medan utvecklingsteamen vill ha snabba leveranser och agila arbetssätt.

**Hur fungerar samarbetet hos er mellan utvecklare och förvaltare? Är dom fortfarande i silotänk där eller?**
Inte mellan utvecklare och förvaltare däremot mellan drift och utvecklare. Det är så organisationen är, väldigt isolerade. Det är till och med så att dom har låst in sig i egna rum och man får inte besöka dem när som helst. Dom ser bara drift av servrar som deras huvudsyssla. De vill liksom inte befatta sig med leveranserna det skall liksom bara fungera.

**Andreas, det här är Nina. En av gruppmedlemmarna**
Ja, hej!

**Hej, jag tänkte det, du sa att DevOps inte är, det är inte ett team utan nåt som hela organisationen skall anamma.**
Ja, som ett koncept.

**Som ett koncept, men är tanken då till exempel på Kriminalvården att man då fortfarande skall ha silon eller hur ett samarbete tänkt fungera om man inte jobbar ihop i ett team?**
Jag vill ha med alla hela tiden.

**Okey.**
Jag vill inte ha några silon som det är idag. Vi har lyckats lite grann vid själva leveranstillfället att få med driftpersonal så de vet ungefär vad det är vi levererar och vad det kan tänkas påverka men det är inte mycket mer än så. Det är väldigt lite i det dagliga arbetet.

**Så det är, skulle du säga att det är att man inte delar med sig av information eller kunskap?**
Ja, så är det, absolut. Och dom vill inte ens ha informationen och kunskapen känns det som för dom tycker att dom är så ”chokade” med de dom håller på med så dom har inte tid med det som vi sysslar med.

**Okey, okey, men skulle du säga att, för någonstans, det verkar vara motstånd men det kanske finns vissa personer som ändå är med på det. Vad skulle du säga är liksom karaktärsdragen för dom här personerna som intresserade av att jobba enligt DevOps?**
Det är personer som har utvecklat och gått till drift. Dom är väldigt positiva till att även arbeta med DevOps från driftshållet och vara mer delaktiga i leveransen.

**Så dom har kunskaper om båda delarna så att säga?**
Ja.

**Ja, okey. Är det något mer du skulle säga som utmärker dom här personerna?**
Ett högt tekniskt kunnande enligt mig. Och en förståelse och känna ansvar för det som rullar på deras servrar.

**Okej. För det är ju lite det vi kollar på, eftersom hur nu är det ju inte teamen någonstans oavsett när man implementerar DevOps så måste det ju som sagt vara personer som vill och passar in att jobba med DevOps, skulle du ändå säga att alla ändå passar att jobba med DevOps?**
Oj… ja, det tycker jag väl. Om man skall leverera enligt DevOps eller implementera DevOps måste ju alla vara med på tåget. Det går liksom inte att folk sätter sig på tvären. Då blir det inte bra helt enkelt, det är min åsikt.

**Du sa att man sätter sig på tvären för att man inte vill, man är inte bekväm med förändring.**
Nej, precis, absolut. Man sitter på sin kammare och gör sitt. Det har vi ett flertal personer som gör här och vill liksom inte vara delaktiga i helheten. Jag har alltid jobbat med statistikuttag och skall syssla med statistikuttag resten av min karriär ungefär. Om man hårddra det väldigt mycket och jag bryr mig liksom inte om något annat.

**Hur skulle du säga att, om det finns, vi snackar om karaktärsdrag hos vissa individer skulle du sa tekniska förmåga skulle du också säga att vissa personlighetsdrag på vissa personer av dom som är intresserad av DevOps?**
Ja, just den här öppenheten och villigheten att dela med sig av både kunnande och information. Det är viktigt.

**Har du även märkt att faktorer som hur länge man varit anställd på organisationen att det spelar roll? Att man kanske …**
Det spelar stor roll, absolut.

**Ja**.
Det har du helt rätt i, det spelar stor roll.

**På vilket sätt?**
Ja, om man ser till dom som sätter sig på tvären så är det dom som varit anställda länge och har liksom bott in sig i ett hörn.

**Det är vanligt förekommande i litteraturen också. Men det är väl det som vi skulle säga. En annan fråga som vi skulle kunna ställa hur mycket mångfald skulle du säga är det på Kriminalvården som du är på nu?**
Om man ser till kön eller till ålder eller?

**Ja, mångfald hela teoribegreppet skulle jag säga.**
Det är väl inte jättebra, det är väldigt många som varit anställda väldigt länge och dom behöver inte prestera någonting om de inte vill. En uppsägning är i stort sett obefintlig på Kriminalvården. Du kan sitta här tills du går i pension och inte behöva göra någonting egentligen om du inte vill.

**Hur stor roll tror du det spelar i den här implementeringen av DevOps?**
Dom sitter oftast på nyckelpositioner så absolut spelar det roll.

**Ja, okey. Det var nog faktiskt den sista frågan. Du har inget mer som du vill tillägga?**
Nej, jag hoppas inte att jag har förvirrat er för mycket.

**Nej, absolut inte. Nu är ju DevOps i säg ett väldigt förvirrande ämne så det ligger ju verkligen inte på dig**.

Nej, det är ju inte helt glasklart. Man ju implementera DevOps på flera olika sätt. Det behöver ju inte vara att man skall kunna ha ett, att det skall komma från ett agilt arbetssätt och att man skall kunna leverera oftare. Det kan ju likaväl vara att man behöver öka kvaliteten på det man levererar. Så kan man också se DevOps.

**Ja, absolut. Det finns många definitioner.**
Ja.

**Och inte är ju fel så länge det fungerar.**
Nej, precis.

**Andreas, om vi skulle få någon mer fråga, får vi lov att skicka ett mejl till dig då?**
Det är bara att mejla på. Inga problem.

# Appendix 4 – intervju 3

**Organisation:** HiQ

**Intervjuperson:** Sheldon Keeping

**Yrkesroll:** Configuration Manager

**Tid och plats:** 10:00 – 11:00. 29 april 2019, personligt möte, Malmö.

**What is your role or your main activity at the organization?**
I am a consultant but what I would call myself is a configuration and release manager but in the DevOps world. I don't work in the old school configuration management and release management which is very much something like ITIL or some old processes that are still used but they don't really fit with DevOps.

**What is a configuration manager?**
Basically, all the tools and systems and processes that are used in DevOps is what I would do as configuration manager. Basically, set up the pipeline and the way of working from the idea to the actual delivery to production. Delivery to production is the release manager part because you have to manage the releases so that you know what's going on. But other people call it DevOps engineer, CICD expert depending on really what you want to call it but they are all pretty much the same thing. Build and release specialist is another one that I have had it called. But they are all intertwined.

**So, you can compare a configuration manager with DevOps engineer?**
Yeah.

**For how long have you been working with DevOps?**
Really since I moved to Sweden, so from 2012, so seven years. Minus a month.

**How are you, HiQ, working with DevOps?**
It's mostly a consultant company so there are a couple of teams that are working in DevOps type of role. I haven't been at HiQ for that long so I haven't had a chance to work with those teams yet I pretty much came right here (The place he is a consultant at). But I know generally how they do their things and what not. In general, the automated builds, the automated deliveries, automated deployments that sort of thing, agile, I think they are using Scrum, that sort of idea, if that is what you are looking for.

**What is your definition of a team?**
A group of individuals that work together towards a common goal.

**Would you say that there are cross functional teams?**
Yeah, I think everybody that is needed in the team is in the team which is the best way to do it really when done right. Everything when done right is the best way.

**You said there was different teams, is it different DevOps teams for different projects? So, you have maybe one team for one customer and another.**
Yeah, generally it's better to split them per customers so that you don't accidentally do something for the wrong customer.

**So, you are responsible for the entire product or project, would you say that? Is the team responsible for the entire project or product?**
Yeah.

**So, who is included in that team? Is it den operations and development?**
So obviously developers, the management that is needed, scrum master, any of the den operativa verksamheten level as well, otherwise it wouldn't really be DevOps. What I find more and more with DevOps is the den operativa verksamheten is not being separated out by specific people they will start running AWS or azure and then a lot of the operation work that would be den operativa verksamheten team would be done automatically by Amazon or Microsoft. So, you don't have to worry about updating machines and all this. You can just kind of push it to the cloud and let it go.

**How do you set up teams?**
I think that due to the nature of projects at HIQ that a new team was setup and used the devops ways of working but the team wasn't setup in order to do devops.

**When HiQ create this kind of teams, how do they do? Do they just like "Oh you don't do anything at the moment so become a DevOps team", how are they picking the people? What does the process look like?**
Generally, if I'm correct here. They find a project and then they find, kind of the right people that are currently either not somewhere. Obviously, that is a big help. And then a mix of junior and senior people because the juniors ones you want to give them more experience, do more consulting, and the seniors ones, well you need the seniors to teach the juniors and to actually have a good product, so when we deliver something it's good which is really important. But you have to have the both. So, both junior and senior in a team and of course in between you get them. You don't want to just have senior people and junior, intermediet developers are also good.

**Where would you put HiQ in this picture with team structures?**
We would be option D. there is no specific person working within ops on any team at HiQ they are all responsible for delivering the project.

**Does every employee have some kind of skill mapping, so when they create a DevOps team, do they know what type of skill every employee has?**
Yeah, we all have internal CVs that we use for consulting and then I'm not exactly sure what the management has on their end but I think they mostly, we have four or five teams. I think one of them is being made into the other ones right now because one of the managers is leaving. That person and the sales people are responsible knowing what the people in that group do, what they know. With the CVs obviously is the paperwork part, they can write down stuff in as well. Things that aren't on the CV but that are good to know. Like on mine, english only would be a part of the good to know. So even if somebody isn't in the team they can look and see, "okay, this is Sheldon, he speaks english, he does this".

**So, what would you say is the most important thing to think about when you create a DevOps team?**
Willingness to learn. From my part of the DevOps team, this is from when I worked at IKEA and not HiQ but when I looked for somebody personally to work with me, experience is great but I can provide that. But more I want kind of willingness to learn, to try things and to get better and better and to make mistakes, but that is okay. Because we should set up things so

that when you make mistakes the world doesn't blow up. That's probably the biggest one. The one guy we hired is when I was at IKEA to help me, he knew nothing of configuration management or DevOps. He worked a month in a different company and then they had shut down the team so he had one month of experience which is nothing.

**Would you say that that person was a specialist in one area that made it possible for that person to learn or willingness to learn?**
He was a tester before, but we weren't working in test at all. So, he just had the right mindset going in.

**So, he was just curious to learn?**
Yeah, he had some scripting skills, which are good, which is also used in testing of course. That's basically it.

**Would you say that skills could be divided into soft skills and hard skills?**
Yes, you can obviously divide them but you kind of need some of each. Because part of the DevOps, the communication is so key between the team.

**But do you prefer soft skills or hard skills?**
Depends on which kind of role they're filling in the team, at least in the beginning. For example, the release manager, they need to have more of the soft skills, the communication, the speaking, because they need to be able to hold the meeting. Where as your lead developer, he needs to know his shit, excuse the word, otherwise the whole thing won't work because the code won't be any good.

**Did we ask what your definition of DevOps is?**
That is a really good question and depending on who you ask you get completely different answers. So, DevOps to me is, I can draw it on the board for you to the infinity of devops depending on which idea that you want to go to but you can find that online. Is the tool chain from the idea to delivery to the customer public whatever the end user is and how that works through automation and it kind of repeat often continuously. Continuous integration is part of that so continuously putting your code in with everybody continuously sharing. Continuous delivery is also a part so you can be able to deliver your code to a customer and then depending on like, you could also look at continuous deployment which is deploying and then continuous release that are different. You don't really want continuous release; you want a release when you want it but you should be able to do it at any point. But you should be able to deploy to a test environment, development environment or preproduction environment at anytime. So, then we are going back to what is DevOps. I have spent a lot of time on this when I started here because nobody knew what is was, everybody kind of has a general idea. It's a way of working that will provide fast feedback lots of automation, automation is your friend, and continuously improving and deploying.

**So, you would say it's a way of working and not a role?**
No. Configuration manager would be the role that does the DevOps way of working. So, if you actually, I'm gonna draw on the board. So, here is the infinity of DevOps, so if I remember correctly, so we kind of have, don't quote me exact on this, so idea, development. So, you kind of have the infinity here, so it starts in the planning, I'll give you the tools, some tools that we use, that can be used for this. So, you have your jira for your planning and design, confluence a lot of weird design would go there, just don't keep age. We use Jenkins but any build system would work that can hook into action. Design and development, because

you need to have the development. So, this is where the coding happens which is a small part, obviously. But it is in reality, it's probably the most, the two places that the most stuff would happen is these two.

**Which two?**
Development and test. You should spend a lot of time on test and the earlier you do it the better. Plan, design, develop then you build and then you test based on what you build. And then, assuming that the test goes well you go onto release, if it doesn't then you kind of go back to planning and loop in here until it works. In a perfect world you can do it every ten min. Release, the difference between release and deploy. Release is more of a soft thing, it's the jira release. So If you're going to say "we're going to release this stuff "we have our no go meetings or whatever sort of release process you want. And then deploy. Deploy is actually where it goes into production. The trick is that these are seperate, then you have multiple realises without deploying anything. The idea behind this is you don't necessarily need to deploy every time. See if you have an android app, you don't want to deploy three times a week unless your Facebook or one of the big companies I can get away with it. But if you're an EON app or IKEA, if you update three times per week people are going to be pissed off and there are going to delete the app. But you still want to have those releases going to that at any point we find something here that's broken, oh well we already fixed it and have a release, then we just deploy it quickly. Then it's ready to go. So you always have several things ready to go at any point in time. And then of course you have the operate which is kind of the continuous work that is going sometimes the operate won't necessarily be on you to run, with the idea for apps. They weren't on your phone I don't have to manage your phone or your computer but maybe you have a back end that you have to operate that it connects to, probably depending on what you do. And then monitor, so you have to file reports and stuff from, for or from the apps where the back end that you need to keep an eye on so if something goes wrong you can plan it fix it quickly.

**If you look the development and operations, where are they in the infinity loop?**
Yeah, I guess originally if you were to separate DevOps to be two separate teams you would have operations on this side and deploy, operate and monitor this is where they connect. And then you'd have design, plan, design, development, build, test on this side. The idea is that everybody in the team should be able to do everything almost, it doesn't have to be every everything but within a team you should have the people that are doing the monitoring, operating, deploying on the team and so that when they are doing this development if something comes up they would say "This won't work in production" they can just say "Hey guys, no, you can't do that, that is a security risk etc. whatever" they know that these guys don't know. And then of course opposite. So, these guys go to deploy and then they are like "Oh no you have to deploy like this" Because the developers know how to do it, because they have been doing it to their test environment or to whatever they're working on constantly. So, they should know how to do this so it's not like a hand off what it used to be as: "Here is the code deployed refraction" and then the development team doesn't care anymore. If something happens here and you find out in monitoring in your current operations the team will all know about it immediately because it's the whole team and you're responsible for the whole thing as a team. So while you had people on the team, so obviously you have a developer here, this is probably a automated system, it should be, if you're going to have automated plus a manual tester, you should always have some manual as well, so you have the scrum masters and release here for that part. Deployment would be in operations and monitoring can be done by some sort of operations person. But your developers would also know how to do this. So, these people can do that, they can do that with a little bit of training or with one-person kind

of being the main guy and then teach everybody how to do it, so that everybody at least has an idea on how it's going to production. Obviously, you have to have control in your production environment. You have to have that everybody can't have access to the production, or shouldn't. You can, but generally you want to have some sort of gatekeeping on the who can access the production system. But if anything goes wrong at any point in here, the whole team is responsible. So, if something comes up on your monitor and everybody has to drop what they are doing somewhere else and fix it if it's enough problem. If it's a small bug with .01 people per million then maybe we will put it in the back log.

**Would you say that you work like this within HiQ?**
I believe that the teams within HiQ yes would work like this whether or not they would actually call it this (DevOps) and it's generally the way that more modern and HiQ being one of the more modern. This is kind of what they agreed to come to without calling it DevOps. Yeah, it's kind of the natural progression of we don't want to have people manually doing the machines and what not and then they're stuck, let's say that it's 6 months between releases what are they doing for five and a half months? They could be a developer that's also helping the development or the owner of the build, the tests, where I generally live is. I'm not a developer. I can write code but that's not what I am good at. I'm in kind of this, the build, test, the automated part not the manual, the release, deploy and monitoring. The operation I do it as well when it's needed but in general, I try to go kind of stateless. I don't want to have a machine that I need to upgrade. It's just a waste of time. If you do it right you don't have to do that.

**So, would you say that you have a main skill, like this is what I'm really good at but then you have some knowledge of other things, so that you could jump into other roles if you are needed there?**
Yeah, my main skill would probably be the build and release, so those two parts. And then deployment. Generally, do it on the same build system because you want to have it automated so it can be done, a script would be needed to be written. But in general, it better to have those done by the developers. The operation making sure that things are running they can do that and it's not a huge thing to learn. Monitoring, I always monitor because you want to know when something goes wrong. The faster you notice something that goes wrong the faster you can fix it. And then my main skill as the DevOps is that I am the git expert. I've taught a course at my previous consultant company for git and git advance.

**So, this having a main skill and knowing other things, we call it T-shaped employee. Is that something you notice in DevOps teams? That you have for example a developer that is really good at programming and developing but then he's also had knowledge over the other things. So maybe you need someone on deploy then he could jump in and help even though his main activity is developing.**
If you pick the right people for a team you should be able to do that. That is the best way because you don't want anything to be dependent on one single person. So, let's say the operate guy, there is one operate guy, what if he leaves? What if he gets hit by a bus? I don't know what's going to happen, nobody knows. Maybe his house gest hit by a meteor, we don't know. I mean they could disappear and you can't have "Oh well we lost our production system, time to restart!" No, you can't do that. So, it's best to have, you should have a main person or the preferably two if you can afford it and then everybody should have a general idea and learn and spend time and doing all the other bits. Because it's important that everybody knows what's going on because people will be missing, they'll be sick or you never know what's going to happen when you have emergency releases or not everybody will

be able to do everything! Let's say that some developers wouldn't be able to be the release manager for a day, they don't have the soft skills. I find it easier to teach the technical skills than the soft skills because communication, speaking in front of a group of people, leading a meeting, it's really hard to learn those things. You can, but it's really hard.

**How does the process look like when you create a DevOps team? Do you just pick people or how do you do?**
They have to be free, if they're not free they are at a different place. Maybe they're so good that we need them in a team. Then you pull them out, but probably not. In terms of financial I believe the consultant out in the field is worth more than a team in general, depending on how much money you get from the team and how many people need to be on the team. But in general, it's better to have them out in the field and you kind of build up their competence with in-house projects. Which is good because then everybody gets better as a company and then when the project is done or when they decide hey, I'm bored with this, they can move on now when they bring all these new tools and ways working for a different company. Generally, the mindset when they, as a consultant company, when they try to hire people it's more, of course the technical skills are really important but what makes a person good to work at HiQ is more important. When I came down to when I started at HiQ we didn't have a technical interview. One because they didn't have anybody else that did DevOps or configuration management so they wouldn't know how to. But they more on where on a fit, who am I as a person, will I fit in here, am I willing to learn. Obviously, my resume speaks for my technical skills so you can look at that, where I've worked references also that. But it was more who I am as a person which is the key for working in this way or any way. If you're working in as a tightened group, you get along.

**Would you say that is the most important thing to have in mind when you create teams?**
Yeah, probably how they work together. Because if you get a cancerous person in a team the whole team will work worse.

**Is that something you know pre-hand? How, okay we think these people will be able to work together or we try and see?**
Obviously, you have to have an idea, "okay they will probably work good together" but then you kind of have to know what kind of person they are. Certain people can get along with anybody, I am one of those people, it's very hard for me to not get along with somebody. I can get frustrated to hell at somebody but I still get along with them. As long as there is a dialog and there is no like "you're an idiot".

**You don't have to like them; you just have to know how to get along?**
Yeah get along and how to work together. Then be a team player. Because there is a team thing, there is no "I did better than you". While if the team fails, everybody fails. It's basically the same idea with sports. It doesn't matter if you score five goals in football but if the other team scores six you still loose.

**So, you would say that communication is very important within the group. For a person to be able to communicate?**
Yeah because if you can't communicate your ideas, you won't be able to function with other human beings. There are places for people that are super inspired but lack communication skills. There are places for them at companies, at whatnot but one of the things I have known since university is… I don't have the top of marks from my class. I did engineering in Canada. I don't, I know that. If you look at the guys who had top of the marks and look at me… The

one guy specifically that I remember named Henry, when he had to do a project you could see the sweat dripping… Presentation. You could see him, wet, dripping down his face. He had trouble talking in front of people and that sort of thing will very strictly stunt you in terms of your development because there is only so much you can do with a person that can't talk in front of other people.

**That is a bit like courage, that you need to be able to speak up?**
Yeah, speak up, you have to take risks and there will be risks. There are times when you have to be like: "Is this good enough? Can you do anything to make it better? Oh okay, well we have to try it." Then kind of we have done all we could. We planned for the worst, hope for the best. But you still have to make that leap of deploying it to 5 million people or trying something new and failing. There is no perfect way to do DevOps. It depends team to team, person to person, company to company. The trick is to treat the DevOps process like this *points to the DevOps infinity loop*. You plan how you're going to do it in the beginning, then you design it, implementing it, you test it and then maybe you release it to another team. If it's not good you have to go back to the planning board and redesign. The process, the tools, the everything. There is nothing perfect in the world but if you run your DevOps team creation like DevOps, it will work. Obviously, it's a smaller team that's doing the devops team creation than the Devops team and it's probably one or two people that's doing the whole thing.

**So, you talked about having the courage to fail and try things. What consideration do you take to responsibility?**
You have to be responsible for your actions, obviously. Because otherwise… For example, we had a guy when I was working at IKEA. He wasn't responsible for his actions. He was kind of ignoring the rules and processes that we had on the developers and the lead developer and him were clashing. He ended up not taking responsibility so they replaced him because he was a consultant. Consultants that aren't working well you just, bye.

**What do you think about decision making?**
You have to be able to make decisions, because if you don't make any decisions you will be stuck on plan. Forever.

**How important do you think it is? Are there other skills that are more important than decision making do you think?**
Yes, but it's definitely up there. Because if you can't make any decisions you can't be courageous because no decision is the worst decision that you can make. You cannot wait for all the info especially not in this world because the tools and technology, it's constantly changing. Everything is changing in this world all the time. Specifically, the IT development world. There is a new programming language released every week or a new upgrade to a machine every second day probably and you have to kind of make decisions based on what you know and then be able to look back and look at the decision and say: "That was a bad decision. How do we fix it?". Or plan. One of the things I'm planning now is we know that the hardware and stuff that we're operating on isn't going to be the final home but we don't have the other home yet and we still need it. So, I'm going to design and build tools so that they can be moved but the decision is made to build those tools in that way so that things can change. You have to make decisions to be able to move forward. If you don't make decisions you don't move

**So, you have already talked a bit about transparency to learn, that it is important that you want to learn but what do you think about adaptability?**
The whole thing is that you're adapting as you go because if something doesn't work you restart and you have to change it. If you do the same thing over and over again it won't work. It's all ones and zeros. If you do it one time it won't work the second time if you do the exact same thing. Hopefully there's not an error or randomness part of your development or planning. So, you have to be able to adapt. Just like with anything in life because things will come up when you're monitoring, when you're deploying, things will break and you have to adapt and fix. You can obviously plan ahead for a lot of things, especially for monitoring. You can't plan ahead because you never know what's going to go wrong. That's why we're monitoring. If we knew it was going to go wrong, we would fix it so it doesn't go wrong. So, the constant adaptment is kind of built-in in the DevOps pipeline. Because otherwise it wouldn't work.

**How important would you say it is?**
Can we write this down so I can kind of rank them?

**Intrapreneurship, which means that team members should be able to come up with new ideas and implement them, how important would you say that is for a DevOps team?**
This is quite important as without it you will never create new things only ever do small improvements.

**We talked about the continuous activities like continuous delivery and continuous integration. What do you think about that? It's about like if a person knows how to use the tools for these activities. What consideration do you take to these activities like continuous integration? What you said before I think is rather important skills to have.**
Continuous integration is not really a skill but what you do. It's the mindset. Anything should always be able to be deployed at any point in time so that mindset is really important and you have to have… And of course you have to have the tools and the back up and the process to match it because if you can't… If the integration delivery takes three days to do the delivery then you can't do that continuously, because it takes 3 days.

**But is it important for a person to know how to use these tools before they enter the devops team?**
No. In general you start off with that one expert, or two experts if you're lucky. It all comes down to budget of course, because money is what makes the world go around. You have that one expert and they hopefully are the type of person that can teach other people how to use the tools, how to set them up.

**So that one person, so they are the experts but do you also. Is it important for them to be able to teach them or is it like okay they know this, this is the only person we have, we guess we just have to work with it?**
Why, if there's no other option you just got to do that but then you get silos. So if that person is sick that day, you can't deliver, you can't do this thing that is important so there should be a certain amount of willingness to whether or not teach by teaching or just like: "Hey you sit with me for a week and you learn everything that I do." You could do that sort of pair programming but with deployment.

**So, you don't have to be skilled at it but you just have to be willing to teach in some way?**

Yeah if you're skilled at something you have to be willing to help people learn what you do.

**What do you think about technical competence, both backend and frontend? How important is that as an ability for a DevOps team?**
Obviously, you need to have those skills if a team needs both of those aspects but not all teams have both

**So, management was operations like that you have to know how to use Linux.**
One of the easier ones is the: “Do you know how to upgrade Linux” or like how to do these things. Most people know how to do it. Whether or not they do it in a good way or not is a different story. I feel like the personality one is teamwork. Because if you don't have the right personality then you can’t work as a team. Anything that's not the hard skills is related to personality. I'm probably going to end up with the hard, like the technology ones near the lower one because if you can learn you can be taught about the tools and the tools, there's so many different options that you can't know them all so probably you will have to teach somebody about a specific tool.

**To summarize you mean that more of the soft skills are more important than the hard skills because you can learn the hard skills with the help from soft skills.**
Yeah, it's harder to teach soft skills. It's easier to teach… Like I can teach Git. It's a really hard thing to learn, but I can teach it. I can’t teach you how to talk in a group or how to adapt under pressure, that's kind of one of the things that you have to learn on your own more than like how someone behaves under pressure, do they just give up and leave or do they hunker down and go for it. I think that's why for me I put the soft skills ahead.

# Appendix 5 – intervju 4

**Organisation:** "IT-huset"

**Intervjuperson:** "Urban"

**Yrkesroll:** Senior Service manager

**Tid och Plats:** 14:00 – 14:50, 29 april 2019, personligt möte, Malmö

**Vi har valt att skriva om i vår kandidatuppsats om DevOps och vi har valt att titta på förmågor hos medlemmarna i teamet. Så det är det grundläggande vi vill gå igenom.**
Ja, absolut.

**Och genom litteratur så har vi liksom identifierat ett par förmågor och kompetenser som dom anser är viktiga i ett DevOps team som vi kommer att gå in på. Men först har vi lite frågor om dina erfarenheter och kunskaper om DevOps. Och då undrar vi vad din roll eller vad dina huvudsakliga arbetsuppgifter på företaget är?**
Man kan säga såhär, för närvarande så ingår jag i ett DevOps team där jag har ansvar för de verktyg som vi använder och kommer att använda inom DevOps. Sen sysslar jag med en del andra saker inom service management också men det kanske inte är så viktigt för det här sammanhanget så att säga.

**Hur länge har du arbetat med DevOps?**
Ja, i ett och ett halvt år kanske drygt nu.

**Skapade ni nya team vid implementeringen?**
Ja delvis, i vissa fall fanns det redan team eller början till. Där det behövdes nya så sattes det upp.

**Hur definierar du "team"?**
De personer/resurser som är allokerade för att utveckla och underhålla en viss tjänst.

**Bilden representerar 4 olika alternativ hur företag jobbar med DevOps mellan avdelningarna. Vilken bild skulle du säga passar bäst in på er?**
B passar nog bäst, vi kommer att sätta upp ett CoE (Center of Exclence) som facilitator

**Hur jobbar ni med DevOps? Hur ser det liksom ut? Har ni tvärfunktionella team eller har ni team inom avdelningarna eller?**
Så du menar du?

**Jag menar typ mellan development och den operativa verksamheten.**
Vi jobbar ju, det finns väl både och kan man väl säga. Vi har ju, eftersom vi inte hållit på med DevOps så länge så finns det egentligen hela skalan, från dom där du har team, som är typ DevOps till dom teamen som nästan står vad är DevOps? Vad är det bra för? Varför skall man ha det? Så att vi har hela skalan faktiskt så det är och det beror ju på den bakgrunden vi har. Behovet har varit olika i olika team och det har funnits olika drivkrafter som drivit fram det här inom, ja man kan säga att vissa team har varit mer intresserade än andra så kan man uttrycka det. Och att behovet har varit större där också såklart.

**Hur ser processen ut när ni skapar ett team? Och då tänker jag på hur ni sätter ihop teamet med dom individerna som skall vara med. Går ni på dom som vill vara med eller är det någon som är ledig så stoppar man in den. Eller utgår ni från vissa kvalifikationer?**
Hm, ja, höll jag nästan på att säga. Men så kan man ju inte svara. Det beror väl lite på men man kan väl säga såhär, det finns ju oftast ett, vad skall man säga, ett core team som är EON sen jobbar vi väldigt mycket med konsulter. Vi har mycket konsulter. Vi är kanske 20–30 procent anställda och resten konsulter någonstans där skulle jag gissa. Och då sätts ju teamen ihop utifrån att man har någon som är liksom är ansvarig för en tjänst eller produkt och det är ju oftast någon som är anställd då som har det. Sen sätter man upp ett team med ja, i huvudsak konsulter eller har man tidigare jobbat kanske med ett visst område. Det är väl snarare så att man försöker titta på kompetenser på dom konsulterna man tar in för att de skall helst kunna så att säga ett agilt arbetssätt, och kunna i alla fall delar av DevOps och dom verktyg som man använder och förstå hur man jobbar i ett sådant sammanhang. Men det är väl liksom en process att vi håller på att transformera hela organisationen till någonting annat än vad det har varit tidigare som är mer DevOps:ig och mer agil.

**Hur går den transformeringen liksom? Har ni stöd från ledningen då eller hur?**
Ja, det har vi ju, eller för DevOps finns det en styrgrupp och för ett nytt arbetssätt finns det också en styrgrupp så att säga som vi rapporterar till hur det går. Man kan säga att ända tills i början på det här året var det väldigt, väldigt få, det var väl typ jag och någon enstaka till eller någon halv resurs som jobbade just med DevOps-delen. Så vi har inte haft massor med resurser men nu har vi liksom försökt göra en satsning från och med i år när vi är lite fler i alla fall så att vi verkligen kan hjälpa teamen med det dom behöver, kring både hur man jobbar och vilka verktyg man skall använda.

**Jag tänker sedan när ni skall anställa eller ta in konsulter, tittar ni bara då på kompetenser utifrån, de skiljer i litteraturen på hårda kompetenser och mjuka typ. Där hårda går mer på kunskap liksom vad dom kan och mjuka mer på förmågor och kompetenser. Tittar ni mer på deras tekniska kunskaper då? Som ni behöver eller går ni mer på?**
Jag tror att man börjar där, när man gör sitt urval och sedan träffar man dom personerna och intervjuar dom och så vidare då kommer mer den sociala biten in och dom mjuka delarna på ett annat sätt. Men utgångspunkten är nog dom så att säga hårda eller tekniska kompetenserna.

**Hur skulle du ställa dem mot varandra, den hårda biten eller den mjuka, vilken tycker du är den viktigaste?**
Ja, det är…

**Ser du den hårda biten som någonting man kanske kan lära sig eller?**
Det kan man absolut göra, det kan man absolut göra så att säga. Jag är väl någonstans men det är väl min personliga uppfattning att under någonstans, femtio femtio liksom, det är också jätteviktigt att man fungerar, både vårt nya arbetssätt och DevOps bygger ju på att man jobbar mycket mer som team så att säga, det är ju liksom teamet som äger problemen eller frågorna hur man nu skall göra eller arbetsuppgifterna, det är inte individer så att säga.

**Är det viktigt för er att kompetensen finns hos alla medarbetarna i teamet, vi säger nu ifall någon försvinner är det viktigt att alla skall kunna?**
I en drömvärld ja, men så är det inte i verkligheten skulle jag vilja säga. Det går inte att få till, det är alltid någon som inte kan allt. Vad man kan observera är ju att, man brukar prata om,

jag vet inte om ni vet vad det är, men man pratar om i-kompetens och T-kompetens. Trenden är väl, eller trenden och kraven från oss och från att säga industrin eller vad man skall säga är väl att man har färre i-människor och fler t-människor som är lite bredare och kan hoppa in och göra en annan arbetsuppgift än att bara vara utvecklare till exempel. Det är väl det vi efterfrågar som arbetsgivare. Att vi vill de personerna är mycket mer värdefulla och det vill säga man är väl beredd att betala mer för de så att säga. Det tror jag är en klar trend att man måste, när vi pratar DevOps med en del team så kan man väl känna att det finns personer som känner sig hotade litegrann av det här för ”jag är ju bara anställd som utvecklare, måste jag testa också? Ja, men det vill jag inte”. Då kanske du skall göra något annat. Nej, men jag kanske överdriver lite men det är påväg åt det hållet. Vi vill ha människor som kan ta olika roller i det här teamet i olika situationer så att säga eller ”jag utvecklar bara, jag supporterar inte saker till exempel är också en sådan fråga som dyker upp ibland. Jag vill bara göra nya grejer, jag vill inte underhålla det gamla typ men det ingår ju liksom i arbetsuppgifterna för ett DevOps team.

**Det är väl det här med förändringsbenägenhet.**
Ja, ja, visst är det så!

**Vilka förmågor och kompetenser skulle du säga att du tittar efter nu om du skall anställa någon eller ta in någon konsult om vi nu tittar på dom mjuka delarna? Vad är det du liksom skulle vilja ha hos en kandidat? Om vi bortser från det tekniska.**
Ja, om vi bortser från det tekniska, jo men det är det här med team, teamkänsla, inte vara rädd för att hoppa på en sak som man inte är helt komfortabel med eftersom det är ett team och någon är borta en dag så måste man kanske gå in och göra någonting som man kanske inte gör varje dag men ändå kunna gör det. Våga gå utanför sin box. Det är väl sådana saker, sedan är det väl saker kring kommunikation. För det handlar mycket om kommunikation om ett team skall fungera så att säga så det är väl det.

**Hur viktigt skulle du säga att kommunikation, kommunikation är en utav de här förmågorna som vi genom litteraturen har identifierat som viktig. Så det är bra att du nämner den. Hur** viktigt **skulle du säga att kommunikationen är? Vi har en lista här efteråt som du skall få gradera viktigheten av de här men om vi bara diskuterar runt omkring dem så länge.**
Nej, men det är väldigt viktigt att man kommunicerar så man liksom får någon teamkänsla. För kommunicerar man inte är man inte en del av teamet heller så att säga.

**Skulle du kunna dra en parallell mellan kommunikation och en annan förmåga som är öppenhet för att lära sig nya saker? För det var du också lite inne på, att man inte kan vara i sin egen…. Vad tar ni för hänsyn till just den kompetensen? Öppenhet för att lära sig nya saker, det var viktigt.**
Ja, det är också viktigt men kommunikationen är väl i så fall viktigare tror jag än just det. Men jobbar man som ett team så måste man också vara beredd att hjälpa andra eller komma in och ta andra uppgifter vid olika tillfällen eller se att här är något som måste göras, men det kanske jag kan göra och hjälpa till att få det att hända så att säga.

**Och då, har du även täckt in lite anpassningsbarhet skulle jag säga. Vilken hänsyn tar ni till beslutsfattande?**
Jag skall väl säga så här, i den perfekta världen så vill väl vi ha så att säga vad heter det, typ autonoma team i princip som kan ta dom allra, allra flesta besluten helt själva och göra sin egna prioriteringar utifrån de krav som ställs på dem. Det är väl någon slags målbild sen är det

väl inte aldrig riktigt så i verkligheten, att man kan det. De flesta besluten skall kunna tas i teamet utifrån den budgeten och så som man har och så vidare. Det skall inte behöva upp i någon hierarki innan man kan göra saker för då tappar man också hela idéen med agiliteten och DevOps. Arbetssättet är ju att man skall kunna göra så att säga mer per tidsenhet. Det är ju lite det man vill uppnå så att säga och skall man då springa runt och fråga får jag göra det här eller inte och så vidare så tappar man liksom, tappat finessen med det.

**Hur långt har ni kommit på den resan där tycker du?**
En bit. Jag vet inte hur man definierar det. Det är återigen som jag sa, vi har team som kommit ganska så långt och andra team som knappt har börjat så att men vi har väl jobbat en hel del med något som vi kallar medledarskap, det vill säga man har försökt att ta bort rätt så mycket av det här att gå och fråga varje gång man skall göra någonting utan man liksom lite mer tillåtande och låter så att säga team eller individer ta sin egna beslut. Men hur långt, hur långt är ett snöre? Det är jättesvårt att svara på.

**Ansvar, vad säger du om den kompetensen i ett DevOps team?**
Den är ju också väldigt, väldigt viktig såklart, för om man skall fungera i team så måste man kunna ta ansvar för det man skall göra. Ansvar betyder ju inte alltid att man kanske gör det själv då kanske man får tala om att nej, jag hinner inte göra det här, kan någon hjälpa mig och så vidare. Det är ju också en form att ta ansvar så att säga. Skall kedjan fungera så måste, den blir ju aldrig bättre än den svagaste länken så att säga så att man. Så den är absolut viktig.

**Mod har vi också, den var du kanske lite inne på. Att man, hur ser du på mod? Och med mod menar vi då att medlemmarna i teamet skall känna sig trygga med att kommunicera men även om dom gör fel skall det inte vara hela världen liksom. Hur ser ni på det?**
Ja, det är ju också en del av det här. Man kan väl säga också om man lyckas implementera DevOps på ett bra sätt så ger det också möjligheter att vara lite mer modig om man säger så. Om man har ordning på sin pipelines, sin leveranskedja och man säger så vidare så är det i alla fall i teorin så skall det vara relativt lätta att fixa till om man ställer till med någonting. Och därav så kan man kanske vara lite mer modig och känna sig trygg i den uppsättning som man har och vågar göra saker och prova saker och se om funkar det här, funkar inte det här, kan jag fixa det med en gång och så vidare. Så utifrån den aspekten så. Och sen vill man ju gärna ha, om det är en del av mod vet jag inte riktigt men också att man har personer som i viss mån, hur skall jag uttrycka det, eftersom man kan göra det här på så många sätt, det finns ju givetvis en teori för DevOps men sen finns det väldigt många variationer om vilka verktyg du väljer och hur du väljer att organisera dig och så vidare. I viss mån utmanar det och säger ”titta här, här är mycket bättre här där vi finns nu” och så valde vi någonting här och att man liksom får en dialog kring det också så att man utvecklar det agila arbetssättet och DevOps så att säga så att man inte fastnar, jaha nu är det här som är lådan, jobbar vi på den utan att man tittar en del utanför. Det är väl en viss form av mod kan jag tänka mig. Att man vågar ifrågasätta också det som uppenbart inte fungerar eller om det finns någonting man tycker är bättre.

**Continuous aktiviteter, då syftar vi på continuous integration, continuous development, continuous deployment, hur viktiga tycker du att dom tekniska bitarna är?**
Ja, för mig är det lite, hur skall jag uttrycka det… Lite motorn i DevOps. Att dom som jobbar med utvecklingsdelarna i alla fall att dom verkligen förstår hur det där fungerar och hur det hänger ihop är jätteviktigt. Fattar man inte hur motorn fungerar så har man lite svårt att göra sitt jobb, typ känner jag.

**Och då antar jag att ni arbetar med dom helt enkelt?**
Ja, det gör vi ju. Där det finns har vi ju motorer höll jag på att säga.

**En fråga bara som jag glömde sedan tidigare, om du skulle definiera DevOps, hur skulle det låta?**
Ok, det är en sådan där million dollar question. För mig är det flera olika eller det finns flera aspekter på det tänker jag. En sak är att man är, DevOps är en enabler på engelska, vad heter det på svenska? Möjliggörare för ett agilt arbetssätt. Men det är också ett sätt att säkerhetsställa kvaliteten på det man kan gör om man använder verktygen på rätt sätt. Och då implicit kan man säga då ett agilt arbetssätt och DevOps tillsammans är ett sätt att göra mer på mindre tid, att kunna leverera snabbare. Det är inte ett entydigt svar men just leverans och kvalitet och att vara en enabler för ett agilt arbetssätt. Det är nog det som är viktigast för mig.

**Kunskap om verktyg och teknologier för DevOps, hur ser du på den förmågan? Är det någonting som du känner skall finnas hos personerna eller är det någonting som du känner att man kan lära sig verktyg i ett initialt syfte så kanske inte det är det jätteviktigaste utan man kan lära sig sen eller är det någonting som du gärna ser att de har med sig in när ni skapar ett team?**
Jag tror att dom flesta kan lära sig det här och jag tror att väldigt många idag faktiskt kan det som håller på med någon form av systemutvecklingen eller någonting sådant där, att dom flesta kan det här. Sen är det säkert så att man jobbar på ett företag så har dom de här verktygen och jobbar man på ett annat är det en annan uppsättning verktyg men grundprincipen är ju densamma och nu är inte jag utvecklare eller någon teknisk specialist men jag inbillar mig att det är relativt enkelt att lära sig hur dom här fungerar. Det är eller det är åtminstone vad jag hör. Det är olika svårighetsgrad såklart beroende på om du pratar code repository så är inte det så mycket annat än ett filhanteringssystem men däremot pratar du Jenkins och sätta upp dina pipelines och dina deployer då är det klart då behöver man ha mer teknisk kunskap för att kunna göra det. Så det är lite olika på olika verktyg också. Men det definitivt någonting man kan lära sig relativt snabbt skulle jag vilja säga och jag tror att kunskapen finns hos dom flesta nu som håller på med mjukvaruutveckling numera. Det är liksom svårt att skicka ett CV utan att det står där.

**Sen har vi, nu blir det en rad uppspaltningar. Vi har förvaltning, och med förvaltning syftar vi då på att teamet skall liksom förvalta produkten eller tjänsten med bland annat övervakning och uppföljning och sådant. Hur ser du på det?**
Vi vill ju liksom inte, hur skall jag uttrycka det. Teamet har ansvaret för en tjänst, kan vara vad som helst egentligen och den och jag vill inte se det som jag har det här är förvaltning och det här är utveckling och så vidare att teamet har totalt ansvar för liksom hela den. Sen hur teamet gör i säg så att säga det spelar inte så stor roll för mig för dom har ett totalt ansvar för både nyutveckling och förvaltning, mindre förändringar och allt vad man nu kan hålla på monitorering. Det är lite de här, vad skall man kalla det självkörande teamen så att säga till skillnad mot vad man ofta hade förr, man hade en utvecklingsavdelning och sen hade man en supportavdelning och sen så utvecklande man där och sen skickade man över det dit och i bästa fall berättade man vad det var man skickade och så hoppades man att dom på supporten kunde ta hand om det. Nu är ju liksom alla på samma ställe jag brukar säga, man får ta hand om sin egen skit. Gör man dålig kod får man ta hand om den dåliga koden typ. Jag vill inte skilja på det, det är en tjänst och man har ansvar för en tjänst och sen är vissa saker förvaltning och andra är utveckling och andra är någonting annat. Så länge man sköter sitt ansvar och prioriterar rätt där så tycker inte jag att man skall prata så mycket i dom termerna.

**Testning, nu jag vet inte, ni kanske har automatisk testning eller manuell men här tänker vi mest på att man skall kunna testa sin egen kod typ och se hur den fungerar. Är det viktigt att dom kan det eller det också någonting som man anser att dom skall kunna lära sig? Eller hur ser ni på det?**
Ja, det är väl ungefär som det andra. Det är någonting man kan lära sig och jag tror att det kommer vara så att man, om vi skall ta in en konsult i ett team så är det ju säkert så att man kanske har vissa delar av det här som man har jobbat med och som man är duktig på men kanske inte andra då får man liksom väga det här kan man lära sig eller det här kan man tillägna sig kunskapen om hur det fungerar så att säga när man väl kommer hit så att säga för man kanske vet att dom övriga i teamet är jätteduktiga på det så det kommer lösa sig utan problem så det är, det också en del av teammixen hur den ser ut och hur man bygger upp team med olika kompetenser och sedan får man inom det kunna bredda sig förhoppningsvis.

**Och det agila arbetssättet har vi sedan här men det har du väl någonstans redan nämnt är en grundförutsättning helt enkelt.**
Ja, det kan man …..

**Hade man kunnat jobba med Devops om du körde enligt vattenfallsmodellen till exempel?**
Ja, absolut. Det skulle jag vilja säga, det går ju. Sedan är det kanske fel att kalla det DevOps då men du får ju någon form av hjälp av dom verktygen som du har för att hålla ordning och reda på dina saker vilket man ju kan göra i ett vattenfallsprojekt absolut.

**Jobbar ni med SLA, Service Level Agreement?**
Ja, delvis. I synnerhet mot våra infrastrukturleverantörer kan man säga. Men inte, men det är inte så att vi på IT har ett SLA med verksamheten på den bemärkelsen.

**Är det någonting ni tar hänsyn till sådär?**
Det är inte en fråga som vi pratar om varje dag om man säger så. Om jag uttrycker mig så.

**Någon form utav ledarskap eller coach till exempel. Nu vet inte jag om ni kör Scrum eller vad ni har. Har ni då någon form av scrummaster. Hur viktigt anser ni att det är att det finns en sådan person?**
Jag tror att det är väldigt viktigt för teamets framdrift så att säga, att det är någon som håller ordning på vad som faktiskt händer och inte händer. Och liksom monitorerar det på något sätt så att det, vi har ju. Tidigare har vi haft mycket projektledare så att säga och nu är det mer scrummasters så att säga även om jag vet att det inte är samma sak riktigt. Men det krävs ju ändå någon som håller ihop och driver fram det liksom absolut.

**Intraprenörskap, att medlemmarna ska kunna ta fram nya idéer och implementera dem, vad anser du om den förmågan?**
Jag tror på ganska stor frihet (entreprenörskap) i teamen inom vissa givna ramar. Man kan t ex inte tillhandahålla ett stort antal DevOps verktyg då kostnaden för licenser och underhåll inte kan motiveras. När det gäller den tjänst eller produkt som man levererar är det teamet tillsammans med Produktägaren (verksamheten) som sätter ramarna för hur innovativ man kan vara.

**Är det några andra faktorer som du tycker att vi har missat som du tycker är viktigt när man skapar DevOps team? Någonting annat som du tycker är viktigt liksom eller**

**grundläggande liksom bör vara med eller att man behöver tänka på?**
Det som har visat sig vara lite svårare än vad vi kanske trodde och det kanske beror kanske på att jag jobbat mycket med det också, det är ju liksom vilka verktyg skall man välja. Jag menar, googlar man DevOps får man sju miljoner träffar liksom och googlar man på code repository så får man fyra miljoner träffar istället liksom. Och hur väljer man på ett vettigt sätt. Hur vet att man att det man väljer är användbart eller fungerar för oss och så vidare. Vi har väl använt, jag brukar säga att vi använt metoden google plus trial and error typ. Det är den metoden vi använt för vi visste ju inte, vi var tvungna att prova, funkar det här. Lyssna med någon som varit med ett tag och någon sådär. Jo, men det här har jag använt hos en någon annan kund eller så. Det här verkar fungera. Ja men låt oss prova det då. Jag tror att många sitter i samma frågeställning. Hur väljer jag. Det finns som sagt så oerhört många och det kommer nya hela tiden känner man ju liksom och hur håller man sig liksom up to date med det så är det, det kan jag. Det är en fråga som vi har spenderat rätt så mycket tid på faktiskt. Och gör fortfarande kan jag säga. Nej, men liksom vi är inte heller klara med den bilden.

**Men ger ni dem fria tyglar till att testa nya till exempel om dom hittar någonting, arbetssätt där som känns, får jag testa det. Är det ok? Eller vill ni liksom, säger ni så här, absolut, testa på eller vill ni gå igenom det först?**
Vi säger väl ungefär såhär, även om vi inte riktigt är där men vår tanke är nu har vi satt upp någon slags grundstruktur som vi vet fungerar i alla fall för vissa team och vill man sedan förändra den uppsättningen så tänker vi att det är klart man får testa men sen måste man också komma med någon slags, ja, vad heter det, value proposition eller business case eller något sådant där för att säga varför vi skall byta i sådana fall eller lägga till. Så vi vill ha någon slags eller vi tror på att man måste ha någon slags governance på verktygen. Någon måste hålla ihop det och känna att det här fungerar tillsammans. Fördelen med det är också är ju att då har man liksom en uppsättning av verktyg så innebär det också att de personer som jobbar kan liksom byta team utan att det blir liksom för stora grejer, jaha, ni gör såhär istället och ni använder något annat där och jaha, nu blir det jättekrångligt. Man kan få en bättre rörlighet mellan teamen, man kanske också kan vara tydligare i sin om man skall hyra in konsulter så kan man säga att ni skall kunna det och det och det och det när ni kommer hit. För det är det som vi jobbar med. Så vi ger ju liksom, ja, friheten, kom gärna med förslag och så vidare gör jättegärna det för vi är så pass få personer som jobbar med det här så vi hinner inte liksom inte sitta och scanna av marknaden heller i alla lägen någon annan som hör någonting hos någon så kommer det men samtidigt kan man inte byta varje dag heller och det blir alldeles för vad skall man säga, ha för många olika verktyg som gör ungefär samma sak blir alldeles för kostsamt också. Licenser är ofta billiga på de här produkterna men däremot i förvaltning och underhåll hela den här delen den ser man ju inte ofta att den kostar pengar har någon som administrerar verktygen. Så det kanske är den delen som jag kommer på sådär spontant att det är det som vi lagt väldigt mycket tid faktiskt, mer än vad jag trodde man skulle behöva göra.

**Du sa innan att man mixar team. få olika kompetenser men försöker ni, mixar ni på något annat sätt för att försöka ta i beaktning andra faktorer? Om man väljer ut personer, kollar ni mest på det här kan dom så att säga eller finns det också andra faktorer som att någon är lite mer junior, någon är lite mer senior?**
Ja, kanske i viss mån. Det är som jag sa innan, jag vet inte om det är unikt för xxx men vi är ju ganska få anställda och många konsulter och skall man vara helt ärlig så när man letar efter en konsult så letar man efter den som är bäst till bäst pris om man säger så. Så att det är mycket det som fäller avgörandet och som har rätt personlighet också vidare och hela den grejen såklart. Men det är ju där man börjar någonstans. Så inte så jättemycket skulle jag vilja

säga att vi funderar så himla mycket på det, för vi har, för nej som sagt eftersom vi bemannar våra team på det sättet så blir inte den diskussionen så aktuell faktiskt, nej. Vi har dom anställda vi har liksom.

**Men använder ni de anställda i första hand tänker jag?**
För vissa roller kan man noga säga så att säga. Men vi har ju, det finns ju oftast för varje tjänst finns det typ en produktägare och det är ju oftast en anställd. Det vill vi att det skall vara, inte alltid är det så men oftast. Vi vill ju liksom att våra scrummasters som liksom håller driver hela shopen skall liksom vara anställda men det är liksom inte alltid så heller så att säga. Det finns en önskan att man på nyckelpositioner har anställda och sedan hyr man in efter behov för resten så att säga.

**Ser du DevOps som en roll eller ser du det som ett arbetssätt?**
Det är för mig ett arbetssätt.

**Jag tänker såhär, när ni har konsulter eller jobbar mycket med konsulter, hur säkerställer ni att kunskapen finns kvar när deras avtal löper ut tänker jag. Nu vet inte jag, här är jag novis, jag antar att man har dem över projektets gång tänker jag. Men det skall ju förvaltas i efterhand. Hur säkerställer ni att den kunskapen finns kvar i teamet?**
Ja, inte som vi borde i alla fall typ känner jag. Det finns inte en uttalad jättetydlig process för hur vi gör det så är det. Man, jag skulle ärligt vilja säga att vi hoppas att inte omsättningen är för stor därav så finns det alltid någon kvar som kan så att säga. Och sedan hyr vi ju in oftast är konsulter här ganska så länge, de är inte här två månader eller tre månader utan det är liksom halvår, år och flera år så att säga. Så vi byter ju inte för varje nytt projekt om jag säger så. Utan det kan vara ju vara så att någon konsult har suttit i tre år på samma team så att säga så det. Men frågan blir ju aktuell när folk slutar. Det man gör oftast är att man säger jaha, nu skall du sluta men okej, då måste du överföra din kunskap till den här personen så att säga. Det kan ju ibland vara en annan konsult faktiskt för att vara helt ärlig. Har man den set upen som vi har är inte det här så lätt att hantera. Sedan finns det givetvis någon slags krav eller grunddokumentation kring produkten som sådan såklart. Och det är i sig är en slags kunskapsöverföring också för man kan läsa sig till. Men jag skulle också säga att jag tycker nog att genom att köra DevOps så kan man ju faktiskt få en del på köpet litegrann genom att man använder dom verktygen som i alla fall delvis skall skapa en dokumentation om vad man har gjort och hur liksom när skickade man liksom den här koden till produktion typ eller när gjordes den här ändringen. Det ger ju verktygen per automatik. De är ju inget som någon behöver dokumentera på sidan så att det är tror jag DevOps kan vara en hjälp faktiskt om man är van vid verktyget så tror jag att det är ganska lätt för en utvecklare att komma in och se, jaha, ok, det är såhär det ser ut, så här ser er pipeline ut, varför man har gjort den scriptningen som ni har gjort och så vidare liksom. Så jag tror faktiskt att det kommer underlätta kunskapsöverföringen.

**Här är dom kompetenserna som vi har gått igenom, jag tror att det är fjorton stycken om jag inte missminner mig. Om du skulle rangordna dom i viktighet från ett till fjorton, där ett är den viktigaste och fjorton är den som är minst viktig.**
Det var inte så lätt. Jag tror ändå att man skall, jag tänker att ett agilt arbetssätt är liksom en grund för allt ihop. Så det är nog den första, att man har en slags förståelse för vad det är och varför man gör det så att säga för annars kan man inte liksom inte förstå DevOps typ. Sen tror jag kommunikation är nummer två. Öppenhet för att lära sig tror jag är nästa. Då tror jag att det är nästa för mig, continuous aktiviteter. Sen tar jag mod tror jag är nästa. Personlighet kan

man ju väva in mycket i. Sen tror jag nog kunskapen om verktygen är något. Testning tror jag är nästa. Då tror jag att jag sätter anpassningsbarhet på nästa, och sedan ledarskap/coach och sedan beslutsfattande. Då tar jag organisatoriska faktorer, och sedan förvaltning och sist SLA.

**Jag känner att det var det som vi hade.**
Spännande!

# Appendix 6 - intervju 5

**Organisation:** inRiver

**Intervjuperson:** Pierre Stehagen

**Yrkesroll:** DevOps

**Tid och plats:** 13:00 – 13:35, 7 maj 2019, personligt möte, Malmö

**Vi börjar med lite inledande frågor om DevOps och sen går vi in på lite mer förmågor. Så vi börjar med… Vad är din roll eller dina huvudsakliga arbetsuppgifter på företaget?**
Det är så där vardagliga som alltid går i ett… Jag tar hand om bygge och då deployer av ny kod. Så, att så snabbt som möjligt få ut färdigutvecklad kod eller features ut i produktionen helt enkelt. Jag ser till att det sker helt enkelt. Automatiseringen. Människor ska inte behöva göra så mycket. Tryck på en knapp och så ska det vara löst helt enkelt. Så ser jag till att knappen finns och knappen funkar.

**Hur länge har du arbetat med DevOps?**
Jag kommer ju från testsidan egentligen och sen så började jag väl med devops för typ 5–6 år sedan när jag satt på Sony Mobile då och de jobbade mycket med det att… Ja men alla skulle liksom kunna koda, bygga, deploya själv och i och med att jag var testledare så tog jag mer hand om det för att jag hade den tiden och sen så har de faktiskt bara fortsatt i den rollen. Det var något jag trivs väldigt bra i. Så då såg jag till att kunna det och var man bra på det fick man mer ansvar och mer jobbmedel. Här är jag helt enkelt.

**Du är på CDON nu eller?**
InRiver heter dem. CDON var min förra så jag började på InRiver i augusti.

**Hur ser processen ut när ni skapar ett DevOps team?**
Vårt team är jag. Nu kommer jag rita här istället. Gammalt klassiskt som ingen egentligen jobbar med idag var ju att man har Dev, man har QA som är mjukvarutest, man har Ops, Den operativa verksamheten som tar hand om hela miljön. Utvecklarna utvecklar någonting. Säger till den operativa verksamheten lägg den här på testmiljön så att de kan börja testa. De hittar en massa buggar, testarna som skickar tillbaka till dem och säger fixa. Så går det att man kastar liksom kod över väggarna och då är… DevOps blir ju liksom den här mitten cirkeln som liksom pratar med alla så, så fort utvecklarna är klara har man continuous integration som kommer in i en given branch. QA, antingen så har man CD, continuous delivery eller deployment, beroende lite på, så att testarna kan testa direkt och när dem godkänt det och det är redo för produktion då tar man det vidare till produktion och då brukar det vara den operativa verksamheten som tar över helt enkelt. Så det är liksom limmet mellan dem här tre som snabbar på hela processen. Så det var ju det dem, inRiver och CDON hade identifierat att de hade ju problem med den här processen och kommer jag som: “Det här är kul, det här älskar jag liksom så det här tar jag jättegärna.” Utvecklarna brukar inte vara så pigga på det för de vill mer koda nya features. Då när jag var på testsidan var det inte många som var så tekniskt kunniga utan de vill liksom testa helt enkelt och inte veta hur det ser ut på baksidan för det förstörde lite hur testningen… Och den operativa verksamheten brukar bry sig mer om andra saker. Nätverk, säkerhet och liksom det. Men här behöver man kunna lite om allt.

**Men är de fortfarande teamen kvar eller är ni?**
Teamen är kvar på… På inRiver är de kvar. På CDON när jag lämnade det så brukar man ha krossfunktionella team som det heter så att varje team har sin utvecklare, sin testare, sin DevOpsare helt enkelt som är intresserad av det. Så att varje team ska vara mer självständigt. Det är nästa steg kan man säga på inRiver också att… Ja varje team ska kunna spruta ut sin egen kod snabbt, smidigt, lätt. Det ska ju inte vara en sån där central roll heller helt enkelt.

**Vad tycker du är det viktigaste att tänka på när man sätter ihop DevOps team?**
Det ska ju funka med alla. Jag ser ju allting, systemet, hela processen. Jag vill ju bara förenkla. Jag vill bara få ut det så snabbt som möjligt. Testarna vill ju jättegärna ha försiktigt. Vill ju inte få in en massa nya features från utvecklarna. Det har ju dem inget intresse av. Utvecklarna vill ju såklart koda och när de är klara vill de kunna glömma det för att gå vidare på nästa sak. Så det är liksom just att man måste vara social för du pratar jättemycket med alla delar. Måste vara tekniskt intresserad så du är uppdaterad på hur allting fungerar från att utvecklarna (något) ska bygga, tills det deployas till testmiljö tills det… Kommer ändra sig lite för att passa i produktionsmiljö. För det är sällan produktioner och testmiljöer är likadana. Det är lite för dyrt så att säga. Vara sociala, tekniskt nyfikna helt enkelt och uppdaterade på allt och gillar att automatisera.

**Vilka förmågor tittar ni efter när ni skapar team? Nu har du förvisso nämnt ett par, är det någon mer?**
Mycket annat också är att du passar in med de befintliga som finns. Det händer att man får någon som är liksom tekniskt sätt jätteduktig, kan precis det vi söker men vi tror inte personen passar in socialt. Och då vill man hellre inte ha in dem. Men det gäller oavsett tjänst. Det är inte bara DevOps där men just eftersom DevOps är lite mer socialt än annat så du tittar man mer där helt enkelt också.

**Kan man säga att man hellre väljer en person som har den sociala kompetensen och tänker att man kan lära sig den tekniska biten?**
Ja, det skulle jag säga.

**Vi har ju med hjälp av litteraturen identifierat förmågor som de anser är viktiga att ha i ett DevOps team. Och det är de vi vill undersöka nu, vad du känner. Vi börjar med beslutsfattande, vad anser du om den förmågan? Är den viktig?**
Ja, nu får jag ju ta där jag jobbat för att både på CDON och inRiver så skulle jag bygga upp allting från grunden. Och då var det ju viktigt att våga ta beslut, stå för dem, ha en åsikt att: "Så här tycker jag för att." Och sen så blir man ju motsagd ibland och då har man en diskussion så man kommer fram till vad som är bäst här. Men ja, man måste kunna fatta beslut och man kan ju inte bara springa och fråga hela tiden för då får du heller ingenting gjort. Måste vara självgående. Ja, viktigt, för att summera.

**Du kommer sen få rangordna de här förmågorna i slutändan. Vilja att lära sig, vad tar ni för hänsyn till den förmågan?**
Den är också viktig för att man vill alltid få allting bättre helt enkelt och utvecklas. Det kommer ju nya tekniker men ja. Önskvärd mer än viktig ska man väl säga.

**Öppenhet för förändring?**
Den är viktig. För att ja, återigen oavsett om man vill eller inte så allting ändras. Nu pratar vi generellt DevOps va ioförsig. I min nuvarande roll så krävs det. Det är ett absolut krav. Sen

var det som sagt… När jag satt på Sony Mobile då var det liksom… Där var man ju mer en kugge i ett stort hjul och kunde ju inte… Så mycket ändrades inte helt enkelt. Så då var det inte lika viktigt där. För där var ju allting liksom på plats. Nu när man precis har påbörjat den här resan då ändras ju allting vecka till vecka nästan. Så det är nog mer var företaget i sig är i sin egen process så att säga.

**Ansvar. Vad tar ni för hänsyn till den? Och med ansvar menar vi då… Ja att man ska ta ansvar för hela processen eller produktionen utav tjänsten eller…**
Jo, det krävs ju också eftersom det är ju mellan alla stolar höll jag på att säga här nu men… Att så fort utvecklarna anser att de är klar så är det ju lite DevOps som flyttar runt koden. Då har du ju liksom ansvar för att den kommer ut som den ska överallt, till produktion… till och med produktion egentligen där den operativa verksamheten tar över så att… Och sen som jag nämnde innan det här att det ska bara vara att trycka på en knapp för vem som helst att få upp hellre men det går också sönder och då får man snabbt fixa. Då är det viktigare att fixa än ha blame-game men det, det är överallt och alla roller och tjänster så… Men jag tror inte det är mycket… Det är nog inte mycket mer än alla andra roller så att säga. Med ansvar för utvecklarna har ansvar för att utveckla det korrekt. Testarn måste ju verkligen se till att testa allting så att… Visst är det viktigt men inte mer än i någon annan roll.

**Anpassningsbarhet, vad känner ni för det? Och med anpassningsbarhet menar vi då att man ska kunna ta över arbetsuppgifter från övriga teammedlemmar.**
Den är nog mycket viktigare. Framförallt för DevOps för det är trots allt är ju liksom en supportorganisation till alla andra sen får man ju anpassa sig för hur vill… Här kan vi ju fortsätta på den här lilla kladden. Här har vi ju tre styckna Dev team liksom. Och alla vill ju nog inte jobba exakt efter samma mall. Utan då får man ju anpassa allting efter hur det funkar för dem. Testarna vill ju ha sina krav att inte det ska ändras för mycket när de håller på att testa så… Ja anpassningsbara, definitivt, jätteviktig.

**Mod och kommunikation.**
Den är viktig.

**Kommunikation syftar vi då på kommunikation inom teamet så klart och mod… Ja men vi knyter mod till det för att man ska våga göra det liksom.**
Ja det var det jag tänkte också. Egentligen bara våga fatta besluten när du sitter och har problem. Löst det. Du behöver inte fråga och säkerställa med alla utan… Lös problemet sen äger du din lösning till vad problemet nu var och… Och kommunikation, gör du saker, händer saker, ja men då ska du… Kommuniceras ut och som ni ser så pratar man ju med alla. Så kommunikation… Kunna kommunicera är viktigt och modet att fatta beslut också superviktigt, framförallt… Det är också det här nu när vi är i stora förändringar, det händer ju mycket. Återigen om man jämför med Sony där allt var liksom… Det var liksom en stor kugge, där händer det inte så mycket, då kan man ju om man vill liksom bara glida med på ett annat sätt. Men jo, man kommer alltid behöva fatta vissa beslut så det modet måste man ha att man vågar fatta beslut och ansvara för sin lösning helt enkelt.

**Intraprenörskap. Handlar helt enkelt om anställda inom organisationer som beter sig lite som entreprenörer, så de kan ta fram nya idéer och sen implementera dem. Är det nånting man i DevOps ska kunna göra som en anställd?**
Ja, väldigt starkt rekommenderat eftersom man håller på med automatisering mycket. Normalt sätt är så är det många manuella steg som man inte tänker på för att man alltid har gjort det. Och då vill man ju automatisera bort det och då bygger man ju upp något annat. Lite nya

beroenden men allting ska ju… Tanken är ju att det ska bli mycket lättare och snabbare. Man ska definitivt ha ett…

**Man ska alltså kunna göra det, ta den idén och implementera idén?**
Ja, det är optimalt.

**Sen har vi continuous aktiviteter då. Continuous integration, development, deployment och alla dem. Hur viktigt är det att man har dem på plats?**
Det blir väl egentligen det man äger som DevOps. Det är hela CI/CD flödet. CI då som sagt continuous integration. CD kan stå för både continuous delivery och continuous deployment. Skillnaden är att delivery är just när man har manuella steg, testare ska testa så att man automatiskt deployar till någon miljö men sen är det… Krävs det helt enkelt ett manuellt steg för att ta det vidare till den operativa verksamheten. Något godkännande. Continuous deployment är det fritt blås ut. Men då har man automatiska tester så att man ändå litar på all ny kod så att man vet att det är tryggt, det är säkert. Går det igenom alla tester så… Då är allting automatiserat helt enkelt. Och eftersom jag kommer från testsidan så är jag egentligen fortfarande livrädd från det då men… Men jag ser vinsten med det om man. Om man vågar det är liksom… Jag har inte varit på ett ställe där jag har vågat hittills.

**Inte?**
Nej men det är ändå slutmålet att komma dit men… Inte riktigt än. Jag vet att CDON sen jag slutade, de har börjat med det. Utvecklarna tycker det går bra men jag vet inte om jag hade hållit med om jag hade varit där då men…

**Men det är alltså viktigt att teammedlemmar känner till de här aktiviteterna och kan jobba enligt principerna?**
Ja precis. Så det… Och där är det ju kort och gott så fort du är färdig med din grej så mergar du i det till master branschen som det heter och då bara… Automatiskt, allting byggs, blir ett paket, du trycker ut det i produktion eller ja, det byggs och trillar ut någonstans och då testar automatiska tester. Går alla tester igenom så fortsätter paketet vidare till produktion och så är det ute helt enkelt. Allting automatiskt, inga manuella testare någonstans.

**Kunskap om verktyg och teknologier för DevOps. Är det en förmåga som är viktig när man ska liksom komma in ett DevOps team?**
Ja, det är det definitivt. I alla fall de här största. Numera är det typ alla… Jobbar ju någonstans i molnet. Azure som är Microsoft, vi har AWS som är Amazons, Google har sitt och Alibaba, ja det finns hur många som helst. Men så brukar det komma verktyg som liksom passar olika bra med molnen. Microsoft har ju hela sin svit och man har liksom allt från källkod till bygge till deploy till tickethanteringssystem. Azure DevOps heter just det då. Men det finns en hel flora av olika verktyg som man ändå bör känna till vad som är för och nackdelar med de olika och vill man automatisera allting och olika Dev team vill av olika anledningar ha olika inom samma företag så ska man ju kunna ta det tycker jag. Sen så brukar det alltid vara en fördel att ha få antal olika produkter så man vill ju samtidigt göra det till ett och då gäller det att veta vilket som passar bäst för det företaget helt enkelt. Så verktyg viktigt.

**Kunskap om verktyg för programmering, full-stack development, vad tar ni för hänsyn till den här förmågan?**
Den är oviktig. Den är viktigare om man är med i cross funktionellt team för då är det sällan att du 100% är DevOps utan du är en kille med ansvar för CICD-flödet (16:08) och bygger deploy. Men då brukar det vara någon annan som just är full stack som tar hand om back end,

front end. DevOps är mer små automatiserings scripts så eftersom vi jobbar med Microsoft så är det Powershell som gäller annars är det ju olika javascript, eller scriptspråk helt enkelt. Det är inte lika viktigt för vill man ha full stack då är det mer en utvecklarroll man ska söka helt enkelt. Det är en bonus. Men då är det nog mer en utvecklare som kan CICD-flödet(?).

**Sen har vi förvaltning och den operativa verksamheten, och det är väl också det här att teamet ska kunna ta arbetsuppgifter från varandra. Att de ska känna till vad som händer i den operativa verksamheten, dom scriperna och kunna utföra dom.**
Ja det är ju den operativa verksamheten det här tänker jag då och det är ju egentligen det som finns i produktion. Man kommer alltid hitta buggar och spela in just utvecklarna fixas, testas och sen ut i produktion igen. Det bör man ju också kunna det ingår ju i hela flödet helt enkelt, det är inte bara nu utveckling.

**Testning?**
Ja, behöver man faktiskt inte kunna så mycket. Automatisering möjligtvis för det är lite dit man vill nå ju. Man vill ju egentligen faktiskt jobba bort manuell testning. Det är ett delmål så man vill ju få ut det så snabbt som möjligt. Sen ska man ju veta risken med det också. Men automatisk testning är mycket viktigare. Nu har du api testning, last testning, prestandatestning så det är väldigt bra att kunna, inget måste per se för oftast finns det utvecklare som tar det eller så har du en dedikerad testare, automatiserar lite beroende på hur teamet är uppbyggt helt enkelt. Men det brukar inte ligga på DevOps rollen det brukar mer vara "Här finns tester, se till att de körs när vi deployar hit ut" till exempel.

**Hur står det till, automatiska tester kontra manuella tester?**
Det skulle jag säga är lite beror på hur man räknar. För du har ju hela enhetstester där ska du göra oändligt mycket nästan. Men nästa steg som är API tester som skickar anrop eller integrationstester som skickar anrop med allting, därefter kan man påbörja manuella tester men då ska ju det vara, det finns ju en sån testpyramid att det ska bli färre och färre, samtidigt som man då kan automatisera mycket manuell testning men det blir aldrig lika bra som en människa som gör det. Men väldigt mycket billigare och snabbare. Av skälet blir ju att det inte blir lika bra som automatiskt testar alltid samma liksom för då säger han inte samma sak som en människa helt enkelt. Den är nog trots allt lite mindre viktig måste jag säga för det brukar inte ligga på DevOps rollen just. Däremot bygger det på att göra den så är det rent klassat att förstå det så att säga. Den har jag nyttjat mycket och kunnat flagga tidigt för att få in tester och framför allt flaggat "Här har vi inga tester ens, jobba på". Men då blir det såhär jobba på upper management att få dem att prioritera att skriva tester för det brukar också glömmas. Det är bara ut med ny feature hela tiden för att man inte tar hand om de befintliga.

**Jobbar ni med SLA? Service Level Agreement?**
Ja. Det gör vi. Superduper, duper viktigt. Nu sitter inte jag i den operativa verksamheten teamet, dom har hand om den. Men det är också det att vi har ju nackdelen här kan jag säga det som påverkar mig mest är ju att jag har hand om deployerna. Vår kod är inte tillräckligt bra så vi kan deploya utan nertid så hur vårt SLA säger ospecificerat att nertid så eller vår nertid på 99,9% säger då. Men just deployer räknas som att den är planerad och därför inte inkluderad i detta så. Men nu har vi det som fönster här för vi deployar antagligen lite då och då. Ja man vi ju träffa det fönstret men det är liksom, det är rätt lätt automatiskt skulle man säga. Då är deployen det här klockslaget så vet jag att det är klart tills dess.

**Så säg att ni vill komma till tvärfunktionella team kanske är målet. Så ska då teammedlemmarna i det här tvärfunktionella teamet, alla ska veta innebörden av SLA och så vidare.**
Precis. Det kommer bli ett definitivt måste där. Men i regel så brukar man då prioritera att kunna deploya utan nedtid. Så om man ser det så kunde vi deploya på black friday för att vi märkte att siten blev lite trög så vi identifierade, okej vi fixar detta lilla här så blir siten lite bättre. Så ut med den under värsta högtiden utan att kunderna skulle märka det helt enkelt, utöver att det blev bättre efteråt. Det kunde vi inte göra här på (företaget han consultar på nu).

**Hur utvärderar man vilka individer som ska ingå i team? Jag tänker nu att du ska sätta samman det här teamet.**
Ja då får man ju först och främst kolla vem som vill, för det är ju rätt många som tycker det är väldigt tråkiga arbetsuppgifter. Så då vill man ju inte tvinga in dom för då blir det ju kört från dag ett. Sen är det som sagt efter det får man ju ta dom man har, vem som passar ihop och vem som kan komplettera varandra med olika kunskaper som sagt. Jag kommer från testsidan så jag ser mycket mer från testsidan vilken kan vara bra om jag får någon från utvecklarsidan som också jobbar mot DevOps för då kommer vi kunna diskutera mer och samsyn från olika vinklar innan allt annat. Även någon från den operativa verksamhetenssidan som säker kommer med någon annan input. Ha lite mångfald kanske kan man väl säga.

**Men ser du mer då på tekniska kompetenser?**
Nä det är väl även. Jag vill ju få in dom i teamen så det blir väl lite där men sen ska man ju synka mellan varandra så man har en övergripande strategi att det här måste alla göra punkt slut för att ändå ha någon enhetlighet. Sen så kan man ju dela upp det lite. Den mänskliga sidan är ju mer att det ska ju funka med övriga teamet helt enkelt. Så det är också viktigt. Då är det kanske mer okej att man har en mindre social kille eller tjej och så tar man någon annan social som springer runt till alla. Det blir lite mer där hur man delar upp arbetsuppgifterna mellan varandra.

**Vad ställer ni för krav på individerna?**
Vilja utveckla sig är väl nästan det viktigaste helt enkelt. Vilja lära sig. Nu på inRiver så förändrar sig allting väldigt snabbt och därmed måste man nästan gilla förändring och ta in allt det nya och vad som händer är att vi byter lite verktyg och gör om mycket i en hög hastighet, så vi hinner knappt med själva helt enkelt. Just nu är det den viktigaste. Sen så när det har landat så kommer ju något annat som är viktigare helt enkelt. Men det är som sagt egentligen vill vi titta på dom när vi har, eller nu blir det mer för hela företaget. Men vänsta strävar efter att bara se hur bra hade dom passat in i befintlig grupp och att det är liksom, det är inte ett DevOps team, det är inte bara jag utan alla jobbar ju egentligen med alla. Det är mycket spring och prat så att säga för att alla behöver samarbeta. Så det är väl det, viljan att utvecklas. Siktar väl lite mera på sociala personer då men absolut inget krav så att säga, men jag tror det underlättar. Ja nyfikna helt enkelt.

**Är det någonting du tycker vi har glömt?**
Nä jag tycker ni har täckt det mesta. Viljan att vilja bygga små fula saker höll jag på att säga. Nä men just det här du säger att "Nu gör vi någonting här och det här är ett manuellt steg" och det är många som inte tänker på det för de har de alltid gjort. Och det är just "Okej men jag kan snabbt och lätt fixa så att du slipper göra detta manuella steget men du kommer ha jättedåligt egentligen för att, är det en torsdag så går den lite annorlunda så kommer det inte funka. Bara du gör precis som vanligt så kommer det att funka". Och så funkar det jättebra för jag har gått vidare i mitt liv och gör något annat och sen så slutar du och nästa kommer och

gör samma sak igen fast lite annorlunda och då funkar det inte och då måste alla gå tillbaka och "Ja vad var det vi gjorde här och varför?". Det är väl också den att se till att avsluta saker på ett fint sätt. För oftast blir det en quick fix och den är sällan vacker och hållbar. Så det gäller att man efter quick fix göra den maintainable så man kan ta hand om den.

**Definition av team? Hur definierar du team?**
Team tycker man nog borde minst vara två stycken i alla fall. Sen är det ju, jag säger ju DevOps team som jag bara råkar vara ensam i men det är lätt att lägga saker i DevOps facket om man säger så som tillhör DevOps. Då blir det två så är det alltid en förberett eller blivit fler över huvud taget. Vi har nämligen pratat lite om att ta från, vi har massa Dev team någon ska sitta som 10–20% DevOps och får börja ta hand om detta själv helt enkelt. Och då ligger då har vi det så kallade DevOps teamet. Då är det vissa saker som ligger som man har som ansvarsområde helt enkelt. Det är väl egentligen det som jag ser som team. Mer en roll och team blir nästan samma.

**Skulle du säga att DevOps är en roll eller skulle du säga att de är ett arbetssätt?**
Egentligen är det ett arbetssätt men den har väl blivit en roll. Så jag har ju, min titel är ju DevOps. Men jag skruvar lite på mig för det borde vara DevOps engineer, DevOps arkitekt, DevOps specialist. Alltså någonting mer. Men det börjar bli relativt vanligt att man har rollen DevOps bara så det börjar ju bli en roll helt enkelt. Egentligen håller jag inte med. *Nu är inte min chef här va hehe.* Jag tycker det borde vara DevOps ingenjör eller DevOps arkitekt eller något liknande.

**Du sa att du tidigare jobbade med tvärfunktionella team på CDON, och det var enligt DevOps?**
Ja precis.

**Så det är nu du har blivit ensam i ett team?**
Jag var väl lite på CDON i början också för de satte upp allting. Då hade vi inte trots allt krossfunktionellt än. Och sen så bytte vi chef och då blev det all in Spotifymodellen och det är ju krossfunktionella teams. Och då blir ju alla så ja men vi har ska ha tre fyra teams och det är jag som kunde DevOps ja då tillhörde jag två team. Databaskillarna var också bara två men dom såg väl till att teamen, dom kunde sitta lite var för sig fast dom jobbade ändå tillsammans. Det var en liten övergång där som blev lite bättre hela tiden. Men när dom var klara så var jag inte där heller helt enkelt.

**Men på inRiver jobbar ni inte krossfunktionellt?**
Nä. Då har vi tre Dev team. Vi har ett QR team. Vi har ett den operativa verksamhetenteam och sen så har vi ett DevOps team kan vi väl säga.

**Men du sa att du ville komma till ett tvärfunktionellt team.**
Ja blir det mycket smidigare och snabbare. Det kräver också att varje team har sina ansvarsområden. Vår nackdel här på Inriver är att vi kommer från en on-prem lösning så att man har en server och en client eller ett par clienter. Och sen tog vi hela den och kasta upp i molnet sen, en server och typ hundra tusen användare. Vi fick det att funka men det var inte bra, det var inte vackert så att säga. Det är i den resan vi sitter nu. Att försöka bena ut och det är därför vi fortfarande har nedtid när vi deployar som en liten skuld från den tiden helt enkelt och all modern kod kan man deploya utan nedtid helt enkelt.

**Men du har ändå erfarenhet av att sätta ihop teams?**

Ja ja precis.

# Referenser


Balalaie, A., Heydarnoori, A., & Jamshidi, P. (2016). Microservices architecture enables devops: Migration to a cloud-native architecture. *Ieee Software*, 33(3), 42-52, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 29 mars 2019]

Barrick, M. R., & Mount, M. K. (1991). The big five personality dimensions and job performance: a meta-analysis. *Personnel psychology*, 44(1), 1-26, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 17 april 2019]

Birley, S. (1985). The role of networks in the entrepreneurial process. *Journal of business venturing, 1*(1), 107-117, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 5 april 2019]

Brunnert, A., van Hoorn, A., Willnecker, F., Danciu, A., Hasselbring, W., Heger, C., . . . von Kistowski, J. (2015). Performance-oriented DevOps: A research agenda. *arXivpreprint arXiv:1508.04752*, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 25 mars 2019]

Bryman, A. (2016). *Social research methods* (Fifth edition ed.): Oxford University Press.

Callanan, M., & Spillane, A. (2016). DevOps: making it easy to do the right thing. *Ieee Software, 33*(3), 53-59, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 3 april 2019]

Clark, C. E., Cavanaugh, N. C., Brown, C. V., & Sambamurthy, V. (1997). Building change readiness capabilities in the IS organization: Insights from the Bell Atlantic experience. *MIS quarterly*, 425-455, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 10 april 2019]

Debois, P. (2011). DevOps: A Software Revolution in the Making? *Cutter IT Journal, 24*(8), Tillgänglig via: Cutter https://www.cutter.com/journals/cutter-business-technology-journal [Hämtad 7 april 2019]

Donofrio, N., Spohrer, J., Zadeh, H. S., & Demirkan, H. (2010). Driven medical education and practice: A case for T-shaped professionals. *MJA Viewpoint*, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 23 april 2019]

Ebert, C., Gallardo, G., Hernantes, J., & Serrano, N. (2016). DevOps. *Ieee Software, 33*(3), 94-100, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 31 mars 2019]

Erich, F. (2018). *DevOps is Simply Interaction Between Development and Operations*. Paper presented at the International Workshop on Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 7 april 2019]

Erich, F., Amrit, C., & Daneva, M. (2014*). A mapping study on cooperation between information system development and operations*. Paper presented at the International Conference on Product-Focused Software Process Improvement, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 30 mars 2019]

Fitzgerald, B., & Stol, K.-J. (2014). *Continuous software engineering and beyond: trends and challenges*. Paper presented at the Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 13 april 2019]

Frostenson, S. (1990). *Det kompetenta teamet : kompetensutveckling och lagarbete i det moderna företaget*: Complan Productions.

Gallivan, M. J., Truex III, D. P., & Kvasny, L. (2004). Changing patterns in IT skill sets 1988-2003: a content analysis of classified advertising. *ACM SIGMIS Database: the DATABASE for Advances in Information Systems, 35*(3), 64-87, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 5 april 2019]

Gartner, W.B., 1988. "Who is an entrepreneur?" is the wrong question. *American journal of small business, 12*(4), 11-32, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 4 april 2019]

Harrison, D. A., Price, K. H., Gavin, J. H., & Florey, A. T. (2002). Time, teams, and task performance: Changing effects of surface-and deep-level diversity on group functioning. *Academy of management journal, 45*(5), 1029-1045, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 5 april 2019]

Hemon, A., Lyonnet, B., Rowe, F., & Fitzgerald, B. (2019). From Agile to DevOps: Smart Skills and Collaborations. *Information Systems Frontiers*, 1-19, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 10 april 2019]

Hisrich, R. D. (1990). Entrepreneurship/intrapreneurship. *American psychologist, 45*(2), 209, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 2 april 2019]

Hollenbeck, J. R., DeRue, D. S., & Guzzo, R. (2004). Bridging the gap between I/O research and HR practice: Improving team composition, team training, and team task design. *Human Resource Management, 43*(4), 353-366, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 3 maj 2019]

Holloway, M. (2017). *Service Level Management in Cloud Computing. [Elektronisk resurs] : Pareto-Efficient Negotiations, Reliable Monitoring, and Robust Monitor Placement*: Springer Fachmedien Wiesbaden.

Humble, J., & Molesky, J. (2011). Why enterprises must adopt devops to enable continuous delivery. *Cutter IT Journal, 24*(8), 6, Tillgänglig via: Cutter https://www.cutter.com/journals/cutter-business-technology-journal [Hämtad 7 april 2019]

Hussain, W., Clear, T., & MacDonell, S. (2017). *Emerging trends for global DevOps: A New Zealand perspective*. Paper presented at the 2017 IEEE 12th International Conference on Global Software Engineering (ICGSE), Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 7 april 2019]

Hüttermann, M. (2012). *DevOps for Developers. [Elektronisk resurs]:* Berkeley, CA: Apress.

Ingdahl, W. (2016). Alla pratar om devops – men vad är det egentligen? Tillgänglig via: https://techworld.idg.se/2.2524/1.647986/devops-agil-utveckling

Jackson, S. E., May, K. E., & Whitney, K. (1995). Understanding the dynamics of diversity in decision-making teams. *Team effectiveness and decision making in organizations, 204*, 261, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 15 april 2019]

Jacobsen, D. I. (2002). *Vad, hur och varför : om metodval i företagsekonomi och andra samhällsvetenskapliga ämnen*: Studentlitteratur.

Jelphs, K. (2006). Communication: soft skill, hard impact? *Clinician in Management, 14*(1), Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 3 mars 2019]

Judge, T. A., Higgins, C. A., Thoresen, C. J., & Barrick, M. R. (1999). The big five personality traits, general mental ability, and career success across the life span. *Personnel psychology, 52*(3), 621-652, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 6 mars 2019]

Katz, R. L. (2009). *Skills of an effective administrator*: Harvard Business Review Press.

Katzenbach, J. R., & Smith, D. K. (2005). The discipline of teams. *Harvard Business Review, 83*(7), 162. Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 7 april 2019]

Kim, G., Humble, J., Debois, P., & Willis, J. (2016). *The DevOps Handbook; How to create world-class agility, reliability, & security in technology organizations* (Vol. First edition): IT Revolution Press, LCC.

Kinlaw, D. C., & Liungman, C. G. (1995). *Medarbetarskap : att på bästa sätt använda och utveckla de anställdas kompetens*: Studentlitteratur.

Krackhardt, D., & Stern, R. N. (1988). Informal networks and organizational crises: An experimental simulation. *Social psychology quarterly*, 123-140, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 3 april 2019]

Lobosco, M. (2019). LinkedIn Report: These 4 Ideas Are Shaping the Future of HR and Hiring. Tillgänglig via: https://business.linkedin.com/talent-solutions/blog/trends-and-research/2019/global-recruiting-trends-2019

Lwakatare, L. E., Kuvaja, P., & Oivo, M. (2015). *Dimensions of devops*. Paper presented at the International conference on agile software development, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 3 mars 2019]

Major, D. A., Turner, J. E., & Fletcher, T. D. (2006). Linking proactive personality and the Big Five to motivation to learn and development activity. *Journal of applied psychology, 91*(4), 927, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 5 maj 2019]

Melnik, G., & Maurer, F. (2004). *Direct verbal communication as a catalyst of agile knowledge sharing*. Paper presented at the Agile Development Conference, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 8 maj 2019]

Morgan Jr, B. B., Glickman, A. S., Woodard, E. A., Blaiwes, A. S., & Salas, E. (1986). *Measurement of team behaviors in a Navy environment*. Battelle Columbus Labs Research Triangle Park Nc, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 23 april 2019]

Oreg, S. (2003). Resistance to change: Developing an individual differences measure. *Journal of applied psychology, 88*(4), 680, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 18 april 2019]

Overhage, S., & Schlauderer, S. (2012). *How Sustainable are Agile Methodologies? Acceptance Factors and Developer Perceptions in Scrum Projects*. Paper presented at the ECIS, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 13 maj 2019]

Paul, F. (2014). The Incredible True Story of How DevOps Got Its Name. Tillgänglig via: https://blog.newrelic.com/engineering/devops-name/ [Hämtad 9 maj 2019]

Perera, P., Silva, R., & Perera, I. (2017*). Improve software quality through practicing DevOps*. Paper presented at the 2017 Seventeenth International Conference on Advances in ICT for Emerging Regions (ICTer), Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 12 maj 2019]

Peterson, T. O., & Van Fleet, D. D. (2004). The ongoing legacy of RL Katz: An updated typology of management skills. *Management decision, 42*(10), 1297-1308, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 20 april 2019]

Rauch, A., & Frese, M. (2007). Let's put the person back into entrepreneurship research: A meta-analysis on the relationship between business owners' personality traits, business creation, and success. *European Journal of work and organizational psychology, 16*(4), 353-385, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 18 april 2019]

Riege, A. (2005). Three-dozen knowledge-sharing barriers managers must consider. *Journal of knowledge management, 9*(3), 18-35, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 13 april 2019]

Robles, M. M. (2012). Executive perceptions of the top 10 soft skills needed in today's workplace. *Business Communication Quarterly, 75*(4), 453-465, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 13 april 2019]

Rubenowitz, S. (2004). *Organisationspsykologi och ledarskap* (3., [omarb.] uppl. ed.): Studentlitteratur.

Schulz, B. (2008). The importance of soft skills: Education beyond academic knowledge. *Journal of communication, 2*(1), 146-154, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 26 april 2019]

Sebastian, I., Ross, J., Beath, C., Mocker, M., Moloney, K., & Fonstad, N. (2017). How big old companies navigate digital transformation. *MIS Quarterly Executive, 16*(3), 1540-1960, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 4 maj 2019]

Shahin, M., Zahedi, M., Babar, M. A., & Zhu, L. (2017). *Adopting continuous delivery and deployment: Impacts on team structures, collaboration and responsibilities*. Paper presented at the Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 27 mars 2019]

Shakir, R. (2009). Soft skills at the Malaysian institutes of higher learning. *Asia Pacific Education Review, 10*(3), 309-315, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 26 april 2019]

Shamow, E. (2011). DevOps: A Software Revolution in the Making?. *Cutter IT Journal, 24*(8), Tillgänglig via: Cutter https://www.cutter.com/journals/cutter-business-technology-journal [Hämtad 7 april 2019]

Shropshire, J., Menard, P., & Sweeney, B. (2017). *Uncertainty, Personality, and Attitudes toward DevOps*. Paper presented at the Twenty-third Americas Conference on Information Systems, Tillgänglig via: AIS eLibrary http://ludwig.lub.lu.se/login?url=http://aisel.aisnet.org/ [Hämtad 20 mars 2019]

Stevens, M., & Norman, R. (2016). *Industry expectations of soft skills in IT graduates: a regional survey*. Paper presented at the Proceedings of the Australasian Computer Science Week Multiconference, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 10 maj 2019]

Weber, I., Nepal, S., & Zhu, L. (2016). Developing dependable and secure cloud applications. *IEEE Internet Computing, 20*(3), 74-79, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 7 april 2019]

Wettinger, J., Breitenbücher, U., & Leymann, F. (2014). *Standards-based DevOps automation and integration using TOSCA*. Paper presented at the 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing, Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 7 april 2019]

Wheelan, S. A. (2017). *Att skapa effektiva team : en handledning för ledare och medlemmar* (Tredje upplagan, reviderad ed.): Studentlitteratur.

Wiedemann, A., & Wiesche, M. (2018). *Are You Ready for DevOps? Required Skill Set for DevOps Teams*. Paper presented at the Proceedings of the European Conference on Information Systems. Portsmouth, Tillgänglig via: AIS eLibrary http://ludwig.lub.lu.se/login?url=http://aisel.aisnet.org/ [Hämtad 23 mars 2019]

Wiedemann, A. M., & Schulz, T. (2017). *Key Capabilities of DevOps teams and their influence on software process innovation: a resource-based view*. Paper presented at the Twenty-third Americas Conference on Information Systems, Tillgänglig via: AIS

eLibrary http://ludwig.lub.lu.se/login?url=http://aisel.aisnet.org/ [Hämtad 23 mars 2019]

Virmani, M. (2015). *Understanding DevOps & bridging the gap from continous integration to continous delivery*. Paper presented at the Fifth International Conference on the Innovative Computing Technology (INTECH 2015), Tillgänglig via: LUBsearch http://lubsearch.lub.lu.se/ [Hämtad 6 maj 2019]