LUNDS UNIVERSITET
Ekonomihögskolan
Institutionen för informatik

Riskhantering vid mjukvaruut-veckling med NPM

Verksamheters arbete med beroendehanteringssystemet Node Package Manager

Kandidatuppsats 15 hp, kurs SYSK16 i Informationssystem

Författare: David Strömbäck
Hannes Carlsson

Handledare: Odd Steen

Rättande lärare: Umberto Fiaccadori
Markus Lahtinen

Riskhantering vid mjukvaruutveckling med NPM: Verksamheters arbete med beroendehanteringssystemet Node Package Manager

ENGELSK TITEL: Risk management in software development with NPM: How organizations work with the dependency manager Node Package Manager

FÖRFATTARE: David Strömbäck, Hannes Carlsson

UTGIVARE: Institutionen för informatik, Ekonomihögskolan, Lunds universitet

EXAMINATOR: Christina Keller, Professor

FRAMLAGD: maj, 2020

DOKUMENTTYP: Kandidatuppsats

ANTAL SIDOR: 47 (100 inklusive appendix)

NYCKELORD: Riskhantering, NPM, mjukvaruutveckling, tredjepartskomponenter, beroendehanteringssystem

SAMMANFATTNING (MAX. 200 ORD):
Att använda JavaScript är nästintill ett krav vid utveckling av tjänster som ska ha någon kontakt med internet. Eftersom Javascript saknar många grundläggande funktionaliteter behövs ett beroendehanteringssystem som NPM för att inkludera och hantera tredjepartskomponenter med behövd funktionalitet. Arbete med NPM innebär dock ny problematik och medför många risker som är svåra att freda sig från, bland annat säkerhetsrisker där NPM-paket kan användas för att introducera skadlig kod. I denna studie undersöker vi hur företag hanterar dessa risker utifrån modeller för riskhantering vid mjukvaruutveckling samt förslag och best practices specifika för NPM. Detta görs genom semistrukturerade intervjuer med respondenter från fem olika företag med olika förutsättningar. Vi finner att riskhanteringen gällande NPM är något bristande, där företag med formell riskhantering ändå kan behandla risker gällande NPM implicit och på individens ansvar, samt att vissa företag inte har någon explicit riskhantering alls. Vi finner även att det finns en tydlig koppling mellan företagens förutsättningar och kontext och hur de arbetar med riskhantering både generellt och gällande just NPM.

Innehåll

1 Introduktion ... 6
1.1 Tredjepartskomponenter, Javascript & NPM ... 6
1.2 Problem och risker kopplade till NPM ... 8
1.3 Forskningsfråga ... 10
1.4 Syfte ... 10
1.5 Avgränsningar ... 10
2 Litteraturgenomgång ... 11
2.1 Modeller och rekommendationer för riskhantering inom mjukvaruutveckling ... 11
2.1.1 Boehms Spiralmodell ... 11
2.1.2 ISO/IEC 16085 ... 13
2.1.3 Best practices för utveckling med open source & NPM ... 15
2.1.4 Riskhantering, möjligheter och strategi ... 16
2.2 Kärnaktiviteter ... 17
2.2.1 Jämförelse av modeller och förslag ... 17
2.2.2 Övergripande strategi ... 20
2.2.3 Planering av riskhanteringsarbetet ... 20
2.2.4 Identifiering av risk ... 21
2.2.5 Analys av risker ... 21
2.2.6 Prioritering av risker ... 21
2.2.7 Riskarkiv ... 21
2.2.8 Övervakning av risk ... 21
2.2.9 Behandling av risk ... 22
2.2.10 NPM-specifika åtgärder och best practices ... 22
2.3 Riskhantering i agil mjukvaruutveckling ... 22
2.3.1 Informell riskhantering ... 22
2.4 Litteratursammanfattning ... 23
3 Metod ... 25
3.1 Litteratursökning ... 25
3.2 Val av metod för empirisk undersökning ... 26
3.3 Val av respondenter ... 26
3.4 Intervjuguide ... 27
3.5 Genomförande av intervju ... 29
3.6 Etik ... 30
3.7 Transkription och analys ... 31
3.8 Validitet och reliabilitet ... 32

4 Empiriskt resultat och analys 33
4.1 Övergripande strategi och planering 34
4.2 Identifiering, analys och prioritering av risker 36
4.3 Dokumentation, övervakning och behandling av risk 37
4.4 Best practices för utveckling med open source & NPM 38
5 Diskussion 43
5.1 NPM – en nödvändighet trots brister 43
5.2 Övergripande riskhanteringsarbete och bakomliggande faktorer 43
5.3 Riskhantering enligt modell, men ej för NPM 44
5.4 Brist på samsyn 44
5.5 Övervakning av sårbarheter, säkerhetsuppdateringar och automatiserade verktyg 45
5.6 Studiens tillförlitlighet 45
5.7 Källkritik 46
6 Slutsats 47
6.1 Forskningsfråga 1 47
6.2 Forskningsfråga 2 47
6.3 Förslag till vidare forskning 47
Appendix A 48
Appendix B 58
Appendix C 68
Appendix D 76
Appendix E 85
Referenser 99

Figurer

Figur 1.1: Beroendegrafen för det populära NPM-paketet react-scripts 7
Figur 2.1: Boehms spiralmodell 12
Figur 2.2: Övergripande strategi och funktioner i riskhantering enligt ISO/IEC 16085 13
Figur 2.3: Övergripande moment i riskhantering enligt ISO/IEC 16085. 13
Figur 2.4: Processmodell för ISO 14
Figur 2.5: COSOs modell 16
Figur 2.6: Delmoment i etablering av policys för riskhantering enligt ISO/IEC 16085 20

Tabeller

Tabell 2.1: Boehms riskhanteringsplan 12
Tabell 2.2: Jämförelse av modeller för riskhantering 17
Tabell 2.3: Jämförelse av förslag 19
Tabell 2.4: Litteratursammanfattning 23
Tabell 3.1: Koncept och alternativa termer 25
Tabell 3.2: Översikt av undersökningens intervjuer 29
Tabell 4.1: Översikt av de undersökta företagens riskhanteringsarbete 33
Tabell 4.2: Jämförelse av företagens övergripande strategi 34
Tabell 4.3: Jämförelse av hur företag arbetar med identifiering, analys och prioritering 36
Tabell 4.4: Jämförelse av företagens riskhantering gällande NPM. 37
Tabell 4.5: Best practices och åtgärder, samt efterlevnad. 38
Tabell 4.6: Kriterier för granskning av paket. 40

1 Introduktion

I introduktionen beskrivs först bakgrunden och problemområdet, innan forskningsfråga, syfte och avgränsningar presenteras.

1.1 Tredjepartskomponenter, Javascript & NPM

Många problem vid mjukvaruutveckling är återkommande, lösningarna på dessa problem kan med fördel brytas ut i separata komponenter som sedan kan återanvändas, inte bara av de ursprungliga utvecklarna utan också av andra. Att använda befintliga lösningar istället för att utveckla funktionalitet på egen hand medför en rad fördelar, dels finns en enorm potential att spara tid och andra resurser, det går även att argumentera för att koden i populära komponenter (som ofta är s.k. öppen källkod) generellt sett är noggrant granskad och håller hög kvalitet.

Programmeringsspråket Javascript lanserades som en del av Netscape Navigator år 1995 i syfte att möjliggöra mer dynamik och interaktivitet på webben. Fram tills relativt nyligen var Javascript det enda scriptspråket som kunde användas i en majoritet av webbläsare vilket har medfört stor popularitet inom webbutveckling trots relativt begränsad användning i andra sammanhang. Sedan lanseringen av Node.js, som möjliggjorde nya användningsområden för Javascript, har användningen av språket ökat lavinartat till att idag vara världens mest använda programmeringsspråk (DeGroat, 2019; GitHub, 2019). Javascript utmärker sig genom sitt standardbibliotek, d.v.s. den funktionalitet som finns inbyggd i språket, som är relativt begränsat i förhållande till andra språk med utbredd användning. Detta leder till ett ökat behov av tredjepartslösningar för att tillhandahålla dessa saknade funktioner, då det sällan är realistiskt att “återuppfinna hjulet” i varje enskild organisation eller projekt (Yegulalp, 2016). Tredjepartskomponenter används givetvis även för att uppnå annan funktionalitet, som inte nödvändigtvis varit lämplig att inkludera i standardbiblioteket.

Tredjepartskomponenter som används i ett projekt, även kallade beroenden, har ofta egna beroenden vilket ger upphov till en komplex struktur av sammankopplade komponenter i flera led, en så kallad beroendegraf. Detta innebär att det totala antalet komponenter som används i ett projekt kan växa mycket snabbt och oöverskådligt vilket medför att administrationen av dem blir svårhanterlig, t.ex. när nya uppdateringar behöver installeras utan att äventyra befintlig funktionalitet (DeBoer, 2020).

Figur 1.1: Beroendegrafen för det populära NPM-paketet react-scripts innefattar 1381 komponenter och 2927 beroendeförhållanden. Visualiseringen har skapats med hjälp av verktyget "Visualization of NPM dependencies" som är tillgängligt via https://NPM.anvaka.com/

En beroendehanterare (som även benämns med begreppen "pakethanterare" eller "pakethanteringssystem") är ett mjukvaruverktyg som underlättar installation, uppdatering och konfiguration av beroenden (DeBoer, 2020). Komponenter i dessa system är "förpackade" i ett standardiserat format och kallas för "paket". Det dominerande beroendehanteringssystemet inom Javascript är NPM, Node Package Manager, andra exempel för andra språk är Ruby Gems (Ruby), NuGet (Microsoft .NET) samt Maven (Java). De löser i första hand två problem:

1. Utvecklare, dvs användare av paket, behöver ett smidigt sätt att hitta, installera, och hantera beroenden i ett projekt.
2. Skapare av paket behöver en kanal där de kan publicera paket och distribuera uppdateringar till dessa.

Ett beroendehanteringssystem består i regel av två komponenter. En klient, dvs det program som används av den enskilda utvecklaren, samt ett paketregister, som kan liknas vid en databas där paket kan publiceras och hämtas. I fallet NPM finns en officiell klient, NPM CLI, men även Yarn är populär. Yarn använder det offentliga NPM-registret som tillhandahålls av Npm inc som förval (Yarn, 2020). Således är mycket av problematiken densamma.

NPM är idag världens största beroendehanteringssystem, med mer än 1 miljon paket, 11 miljoner utvecklare och över 10 miljarder nedladdningar i veckan. Bland användarna finns såväl hobbyprojekt som namnkunniga företag som t.ex. Microsoft, Netflix, IKEA, Visa (Npm inc, 2020a)).

1.2 Problem och risker kopplade till NPM

NPM särskiljer sig från andra beroendehanteringssystem på en rad punkter. Decan, Mens & Claes, 2017 har kartlagt beroendedjupet hos NPM-paket kontra paket i systemen CRAN och RubyGems. Resultatet visar att NPM-paket har ett större antal direkta beroenden, och ett betydligt större antal transitiva (indirekta) beroenden än de övriga. Siffrorna tyder på att en annan kultur råder inom Javascript-ekosystemet, där utvecklare tycks vara mer villiga att återanvända även små byggstenar. Detta bekräftas av Abdalkareem et al., 2017, som har studerat användning av s.k. "triviala" paket inom NPM-ekosystemet. De fann att 57,9% av NodeJS-utvecklare inte uppfattar användning av triviala paket som något negativt. Detta motiveras främst med att dessa paket är väl genomarbetade och testade samt att användningen av dem ökar produktiviteten. Trots detta framhåller 55,7% av respondenterna att triviala paket komplicerar beroendehanteringen, och innebär en ökad arbetsinsats, bl.a. för att hålla alla paket uppdaterade.

Zimmermann et al. (2019) beskriver en liknande situation, ett genomsnittligt NPM-paket har ca 80 beroenden och 40 olika underhållare med möjlighet att påverka koden. De fann att vissa populära paket är inkluderade i över 100 000 andra paket, samt att de 391 mest aktiva underhållarna har inflytande över 10 000 paket, vilket gör både de populära paketen och dess underhållare till mycket attraktiva mål för en ev. angripare.

Enligt Decan, Mens & Claes (2017) finns ur paketanvändarens perspektiv ingen skillnad mellan ett paket som innehåller en sårbarhet, och ett paket vars beroende innehåller en sårbarhet. En lyckad attack riskerar alltså att drabba samtliga användare av det berörda paketet, samt alla användare av paket som är direkt eller transitivt beroende av det berörda paketet. De flesta attackerna riktar sig mot de personer eller organisationer som använder paketen i sin utveckling, men även slutanvändare kan bli måltavlor.

I en annan undersökning beskriver Decan, Mens & Constantinou (2018b) fenomenet "technical lag" som kan uppstå p.g.a. det stora beroendedjupet hos vissa NPM-paket. När en uppdatering publiceras måste den accepteras av samtliga paket i "kedjan" för att nå slutanvändaren. Fördröjningar i varje enskilt led innebär att det i många fall tar lång tid för uppdateringar att propagera genom ekosystemet vilket leder till att sårbarheter riskerar att lämnas utan åtgärd under denna period. Detta innebär att även övergivna paket utgör en risk då de förhindrar uppdateringar. Undersökningen omfattade 120 000 paket, med 1,4 miljoner publicerade versioner ("releases") och 8 miljoner beroendeförhållanden mellan dessa. Resultatet visade att 1 av 4 beroenden, eller två av fem publicerade versioner led av eftersläpningar. Den genomsnittliga eftersläpningen för dessa paket var 7 till 9 månader. Vidare fann de att den genomsnittliga uppdateringsfrekvensen var ojämnt distribuerad, 25% av alla paket uppdaterades inom en dag, och 25% betydligt mer sällan.
Det är viktigt att notera att NPM tillhandahåller funktioner för att underlätta övervakning av sårbarheter och installation av uppdateringar. En automatiserad granskning kan initieras av utvecklaren med kommandot "NPM audit", men utförs även automatiskt varje gång ett nytt paket installeras. En lista över alla projektets beroenden sammanställs och skickas till paketregistret, som svarar med en rapport över alla kända sårbarheter samt föreslagna åtgärder. Många säkerhetsproblem kan åtgärdas automatiskt (Npm inc, 2020b). Utöver NPM finns ett antal aktörer som erbjuder liknande tjänster, bl.a. Snyk. Även GitHub, som numera äger NPM, tillhandahåller s.k. vulnerability tracking som en integrerad del av versionshanteringssystemet (GitHub, n.d.). Semantisk versionshantering gör det möjligt att specificera beroenden "löst", vilket i praktiken innebär att uppdateringar som inte påverkar

bakåtkompatibiliteten enkelt kan installeras med kommandot “NPM update” utan att projektets konfiguration behöver ändras (Npm inc, 2020c).

Den kanske mest uppenbara risken kopplad till NPM är att skadlig kod kan publiceras i paketregistret, antingen i form av en uppdatering till ett redan befintligt paket, eller som ett nytt paket med ett namn som är vilseledande eller avsett att förväxlas med ett redan etablerat paket, s.k. typosquatting (Npm inc, 2017). Konsekvenserna av ett sådant angrepp kan bli mycket allvarliga och problemet förvärras ytterligare av det faktum att NPM-paket kan innehålla s.k. script som körs automatiskt, t.ex. vid installation. Sådana script kan användas för att angripa inte bara slutprodukten där paketet används, utan även den miljö i vilken produkten utvecklas (Npm inc, 2016).

En attack på ett redan befintligt paket kräver ett konto med behörighet att publicera uppdateringar för just det paketet, en angripare kan bereda sig tillgång till ett sådant konto genom exempelvis kontokapning, social ingenjörskonst eller genom de mekanismer som finns för överföring av ägandeskap i NPM. En sådan attack drabbade paketet eslint i juli 2018, då en angripare kom över inloggningsuppgifter som läckts i ett annat sammanhang, men visade sig vara giltiga på NPM pga. återanvändning av lösenord (ESLint, 2018).

En annan möjlighet är att någon som givits tillgång på legitima grunder visar sig ha ont uppsåt, ett exempel på detta är event-stream-incidenten, där en angripare beviljades ägandeskap av ett övergivet paket. Den nya ägaren föreföll vara seriös och publicerade ett antal uppdateringar, först efter att antal månader publicerades förklädd skadlig kod avsedd att stjäla så kallade krypto-plånböcker från appen Copay. Angreppet förblev oupptäckt i 46 dagar och det är oklart hur mycket användaruppgifter och kryptovaluta som stals (Grander & Tal, 2018).

Sårbarheter i paket förekommer givetvis även utan ont uppsåt, t.ex. på grund av misstag eller andra oförutsedda faktorer. Även legitim avpublicering av paket kan få stora konsekvenser. En sådan incident inträffade i Mars 2016, då en utvecklare avpublicerade över 250 paket p.g.a. missnöje efter att Npm inc ingripit i en pågående tvist mellan honom och meddelandeplattformen Kik. Åtminstone ett av dessa paket var mycket populärt och laddades ner mer än 2,5 miljoner gånger i månaden. Både paketets användare och användare av paket som förlitade sig på det avpublicerade paketet drabbades då det inte längre var möjligt ladda ner paketet från det offentliga paketregistret. Npm inc svarade genom att publicera paketet på nytt utan utvecklarens samtycke (Williams, 2016).

Användningen av tredjepartskomponenter med sårbarheter förefaller vara utbredd, Decan, Mens & Constantinou (2018a) citerar siffror från bl.a. Contrast Security, som visar att 80% av den totala kodmängden i dagens applikationer härstammar från externa bibliotek och ramverk, och ca en fjärdedel av dessa innehåller kända sårbarheter. De konstaterar också att de dröjer mer än 24 månader innan 50% av alla sårbarheter upptäckts. Av 610 000 analyserade NPM-paket visade sig 133 602 stycken vara direkt beroende av andra paket med kända sårbarheter, och 52% av dessa hade publicerat minst en version som använder en påverkad version av ett sårbart paket. Detta trots att det totala antalet paket med sårbarheter endast uppgick till 269 av 610 000. De konstaterar även att mängden kända sårbarheter i NPM-paket tycks öka över tid, samt att en majoritet av dem är av allvarligare art. De understryker även att det med största sannolikhet funnits sårbarheter som ännu ej upptäckts vid tidpunkten för undersökningen. En analys av 430 000 webbplatser, utförd av företaget Snyk, visade att 77% använde minst ett frontendbibliotek med kända sårbarheter (Snyk, 2017).

Trots ett antal uppmärksammade incidenter och den forskning som finns kring dessa saknas undersökningar om hur organisationer arbetar med riskhantering vid användning av språkbaserade beroendehanteringssystem och varför de arbetar på det viset.

1.3 Forskningsfråga

Forskningsfråga 1: Hur arbetar systemutvecklingsföretag för att hantera de risker som kan uppkomma vid utveckling med beroendehanteringssystemet NPM?

Forskningsfråga 2: Vilka faktorer föranleder vald strategi för riskhanteringen?

1.4 Syfte

Målet är att utreda hur företag förhåller sig till de risker som användning av beroendehanteringssystem kan innebära, att undersöka förekomsten av relevanta policys och/eller mer informella arbetssätt, samt vilka överväganden som ligger till grund för dem.

På de företag som arbetar aktivt med kontroll och riskhantering av nya paket och uppdateringar vill vi utforska samt beskriva hur detta görs.

Undersökningen ämnar bidra till att fylla forskningsluckan om hur organisationer arbetar för att hantera de risker beroendehanteringssystem introducerar till mjukvaruutveckling. Problematiken är komplex och spänner över många olika forskningsfält och en eventuell lösning erfordrar en kombination av IT, organisation och strategi.

1.5 Avgränsningar

Uppsatsen avgränsas från beroendehanteringssystem som Maven, Ruby Gems och NuGet, urvalet består istället av beroendehanteringssystemet NPM. Dels för att systemet och dess tillhörande språk, Javascript, är störst på marknaden, men även för att många uppmärksammade incidenter har inträffat på just NPM, vilket innebär att det finns ett omfattande teoretiskt underlag att tillgå. Vi avgränsar oss också från att titta på vilka konsekvenser den valda strategin för riskhantering gällande NPM-paket får. Uppsatsens kommer inte att göra några jämförelser mellan öppen källkod och proprietär programvara, dels för att denna fråga är mycket omdiskuterad och riskerar att skifta fokus från beroendehanteringssystem, men framförallt för att mjukvara som publiceras på NPM nästan uteslutande omfattas av en öppen licens. Vi kommer inte heller att undersöka risker relaterade till licenser och andra juridiska aspekter.

2 Litteraturgenomgång

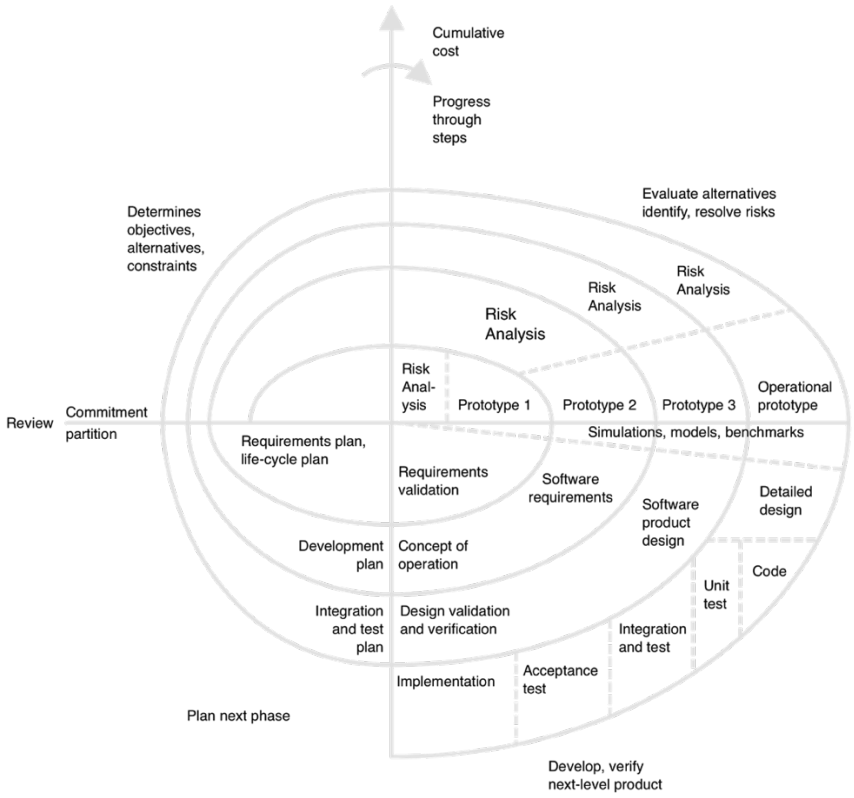
Kapitlet inleds med en genomgång av olika modeller för riskhantering vid mjukvaruutveckling. Det saknas modeller framtagna för riskhantering vid arbete med just NPM, som innebär särskilda utmaningar och risker. Först används dessa existerande ramverk och teorier för att bygga en grundläggande bakgrund om vilka delprocesser som traditionellt finns i modeller för riskhanteringsarbete och för att visa vilka delmoment som i någon form kan förväntas att återfinnas vid god riskhantering. Därefter lyfter vi förslag för hur organisationer bör arbeta med specifikt NPM, vilket tillsammans med aktiviteter från riskhanteringsmodellerna formar vårt teoretiska resultat. Vi inkluderar även ett kortare avsnitt om riskhantering vid agil utveckling, då de undersökta företagen i denna studie i stor utsträckning arbetar agilt är det viktigt att ge en kortare introduktion till vad det arbetssättet kan innebära.

2.1 Modeller och rekommendationer för riskhantering inom mjukvaruutveckling

Här presenterar vi modeller för hur riskhantering kan gå till i mjukvaruutveckling. För att förstå hur verksamheter arbetar med de risker NPM kan medföra, måste vi också ha en bakgrund i hur risker brukar hanteras. Då ingen modell finns framtagen för hur just risker vid arbete med NPM kan hanteras tittar vi här på ett flertal modeller för att kunna identifiera återkommande aktiviteter i riskhantering. Dessa återkommande aktiviteter kallar vi sedan kärnaktiviteter, och är en del av vad vi i undersökningen undersöker ifall företag använder, och hur de aktiviteterna i så fall påverkar arbetet med NPM. Innan dessa modeller jämförs för att hitta dessa kärnaktiviteter, presenterar vi modellerna och belyser centrala punkter per modell.

2.1.1 Boehms Spiralmodell

En av de första modellerna för riskhantering inom mjukvaruutveckling var Boehms spiralmodell (som är en modell hela mjukvaruutvecklingsprocessen) vilken kom som ett alternativ till vattenfallsmodellen i slutet av 80-talet och lade stor vikt vid ett kontinuerligt arbete för riskhantering, där risker identifieras, analyseras, och löses för varje fas. Modellen är central för utvecklingen av riskhantering för mjukvaruutveckling och är därför viktig att inkludera i vår undersökning.



Figur 2.1: Boehms spiralmodell var en modell för hela mjukvaruutvecklingsprocessen, vilken innehöll aktiviteter för riskhantering (Boehm, 1988).

Modellen kan dock menas vara en annan version av vattenfallsmodellen då den fortsatt går i tur och ordning över till nästa fas, men den cykliska utvecklingen med delleveranser samt Boehms fokus på riskhantering och planering är något som fortfarande är aktuellt (Boehm, 1988; Söderland et al., 2012). Modellen som illustreras i figur 2.1 innebär också en tydlig koppling mellan strategi för riskhanteringen och strategin för utvecklingsprojektet, vilket återkommer i senare modeller. Boehm erbjöd en sammanfattad plan för hur riskhanteringen bör fungera, vilken innehöll fem punkter:

Tabell 2.1: Boehms riskhanteringsplan (Boehm, 1988)

1	Identifiera projektets topp 10 risker
2	Ta fram en plan för att lösa varje risk
3	Uppdatera topplista för risker, planer och resultat var månad
4	Belys riskstatus i projektutvärderingar var månad • Jämför med tidigare månads status och rankning
5	Initiera lämpliga tillrättande aktioner

Punkterna i tabell 2.1 finns i någon form kvar i många av dagens modeller för riskhanteringsstrategier, men saknar vissa aspekter och en tydlighet som kan finnas i senare ramverk.

2.1.2 ISO/IEC 16085

I ett samarbete mellan ISO, IEC (International Electrotechnical Commission) och IEEE togs standarden 16085 fram 2004 för hur företag bör hantera risker vid mjukvaruutveckling. ISO/IEC 16085 är alltså en standard, och är därför intressant för undersökningen då den här används för att komplettera akademiska modeller som en mer praktisk modell tagen från industrin. Den är dock fortfarande akademiskt grundad genom samarbete med IEEE. Standarden innehåller rekommendationer för varje steg från analys och planering till implementation och behandling (ISO, IEC, & IEEE, 2004). ISO/IEC 16085 erbjuder många mindre modeller för delprocesser, men två mer övergripande och kärnfulla av dessa mindre modeller är följande:

a) Omfattningen av riskhanteringen bestäms

b) Lämpliga strategier för riskhantering definieras och implementeras

c) Risker identifieras genom en strategi under riskers utveckling genom projektet

d) Riskerna analyseras och prioriteras

e) Verktyg och strategier för att motverka risker definieras, implementeras och utvärderas för att förstå ändringar i status av risk, samt hur väl övervakning av risk fungerar

f) Lämpliga mått tas för att minska eller undvika effekten av risker.

Figur 2.2:. En övergripande modell över vilka typer av strategier och funktioner som bör finnas i ett företags arbete med riskhantering vid mjukvaruutveckling i planeringsfasen (ISO, IEC, & IEEE, 2004).

Figur 2.2 visar att det är viktigt att ha en övergripande plan för hur riskhanteringsprocessen ska fungera, där beslut inte tas isolerat utan som en delmängd av en större strategi. Detta arbete påbörjas tidigt där strategin sedan informerar hur processen fungerar genom livscykeln. 16085 innehåller även en lista av aktiviteter som ingår under projektets gång:

a) Planera och implementera riskhantering

b) Hantera projektets riskprofil (hur villigt projektet är att ta risker, och hur dessa risker kan påverka projektet)

c) Utför analys av risk

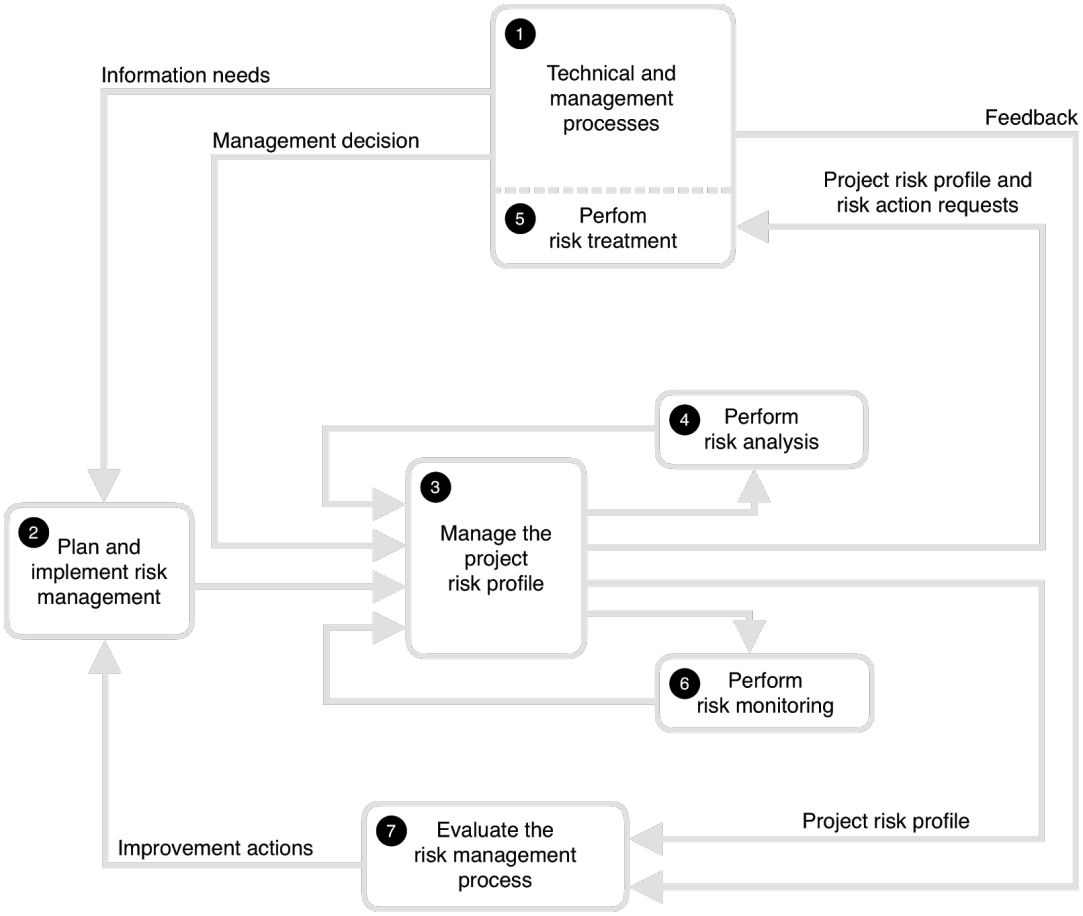
d) Övervaka risker

e) Möt risker (ifall en risk förverkligas, möt den utifrån förutbestämd strategi)

f) Utvärdera riskhanteringsprocessen

Figur 2.3: En övergripande modell över vad företag bör utföra genom utvecklingsprojekt (ISO, IEC, & IEEE 2004).

Denna modell i figur 2.3 är något mer koncis och handlar explicit om hur den faktiska riskhanteringsprocessen bör utformas och vad den bör utföra. Mycket känns igen från föregående modell, då centrala aktiviteter är analys, övervakning och bemötande av risk, samt utvärdering av processen. Identifiering av risker görs alltså i ett tidigare stadie, här handlar det snarare om hur kända risker hanteras. För en bättre översikt av hur processen ser ut erbjuder ISO/IEC 16085 även en enklare modell:



Figur 2.4: Processmodell för ISO. Modellen beskriver den cykliska naturen av hur företag bör arbeta i riskhanteringsprocessen, och hur de olika delprocesserna relaterar till varandra (ISO, IEC, & IEEE 2004).

Figur 2.4 visar att aktiviteter genomförs inte isolerat utan cykliskt i anslutning till varandra. Aktivitet 3, hantering av riskprofil, får en central roll då den dels håller information om risker, men också avgör hur villigt ett företag är att utsätta sig för dessa risker. Den får denna information av aktivitet 4 och 6, riskanalys och riskövervakning. Aktivitet 5, där risker bemöts har inget output till cykeln utan agerar utifrån information från aktivitet 3 (och därmed 4 och 6 också) för att minimera konsekvenser av utfallen risk. Jämfört med riskhanteringsplanen i Boehms spiral-modell (Boehm, 1988) är ISO/IEC 16085 betydligt mer detaljerad och utförlig. Riskprofilen håller här den mesta av informationen och delprocesserna föreslagna av Boehm i tabell 2.1 (då analys och monitorering av risk kan ses som delprocesser av riskprofilen, och den håller information om planer för att bekämpa risker), identifierade och prioritering av risker och kontinuerlig utvärdering av dessa. Steg 5 fyller sedan Boehms punkt (tabell 2.1, punkt 5) att behandla risker. De två modellerna överlappar då de har en mängd gemensamma nämnare, men ISO/IEC 16085 är betydligt mer detaljerad och erbjuder en tydligare struktur.

2.1.3 Best practices för utveckling med open source & NPM

En modell från en ISO-standard innebär inte att den alltid följs i verkligheten, eller att organisationer utför exakt de steg som modellen föreskriver även där den implementeras. Men denna standard är nyttig för att ge en övergripande vy för hur riskhantering i mjukvaruutveckling borde gå till enligt ISO och IEC. Men för mjukvaruutveckling med NPM finns unika utmaningar som varken Boehm (1988) eller ISO/IEC (2004) kunde ta hänsyn till, eftersom NPM inte existerade vid framtagandet av dessa modeller. Därför är det intressant att titta på undersökningar som rör riskhantering i liknande situationer som vid utveckling med Open Source Software (OSS), då NPM-paket i många fall kan bestå av öppen källkod. I en undersökning av riskhantering gällande OSS av Hauge et al. (2010) fann de att ett viktigt steg för att minska risk är att fördela ansvar och att dedikera resurser specifikt för utveckling med open source-mjukvara. Vidare menar de att det är viktigt att utvärdera risker, att utvärdera OSS-produkter och communityn och att ha en tydlig strategi (Hauge et al., 2010). Dessa fynd passar in i ISO/IEC 16085, i steg 2, 3 och 4 i figur 2.4. Det som sticker ut är strategin att utvärdera OSS-produkter och OSS-communityn. Detta sker tidigare i processen vid val av OSS, där gäller alltså inte strategier för hur redan adopterad OSS ska hanteras i ISO/IEC 16085. Denna utvärdering skulle alltså kanske snarare passa in i figur 2.2 eller 2.3 där viss planering finns. En förklaring till denna skillnad kan vara att ISO/IEC 16085 likt Boehms spiralmodell dels är generella och övergripande då de är tänkta att vara tillämpliga för flera olika typer av mjukvaruutveckling, och då saknar viss specificitet. En annan möjlig förklaring är att båda modellerna nu har hunnit bli något föråldrade, då miljön för mjukvaruutveckling idag har blivit än mer komplex än då de publicerades exempelvis genom framtagandet av NPM. Undersökningen av Hauge et al. (2010) är intressant då den tar hänsyn till risker specifika för utveckling med OSS, vilken utveckling med NPM kan menas vara en underkategori av. Men många av lärdomarna är överförbara, som att utvärdera OSS-produkter och communityn, samt att ha en uttryckt strategi för hur arbetet med just OSS ska genomföras.

Det saknas publicerade studier om hur användare av NPM kan skydda sig. Både Duan et al., 2020 och Vaidya et al., 2019 kommer med ett antal förslag på åtgärder som kan vidtas av paketanvändare och utvecklare, men dessa studier är nya och har inte hunnit genomgå peer review. Vi anser dock fortfarande att det kan vara motiverat att inkludera deras råd i syfte att jämföra huruvida de är förenliga med redan existerande riskhanteringsteori, samt att undersöka om de redan tillämpas i branschen. Förslagen innefattar bland annat:

- **Användning av privata paketregister med av organisationen kända versioner av paket** (Duan et al., 2020)**.**
  Detta kan beskrivas som en form av riskbehandling, avsedd att minimera risken att förlora tillgången till paket som används i projektet. Ett sådant paketregister underlättar även centraliserad kontroll, då nya paket måste genomgå en granskningsprocess innan de kan användas av en enskild utvecklare.

- **Regelbunden kontroll av nya säkerhetsråd** (Duan et al., 2020)**.**
  Denna åtgärd kan liknas vid "övervakning av risk", genom att hålla sig ajour med nya säkerhetsrapporter kan risker upptäckas och hanteras innan de får allvarliga konsekvenser.

- **Skyndsam installation av tillgängliga uppdateringar** (Duan et al., 2020)**.**
  Även detta är en form av riskbehandling, genom att hålla sina paket uppdaterade minskar risken för obehandlade sårbarheter.

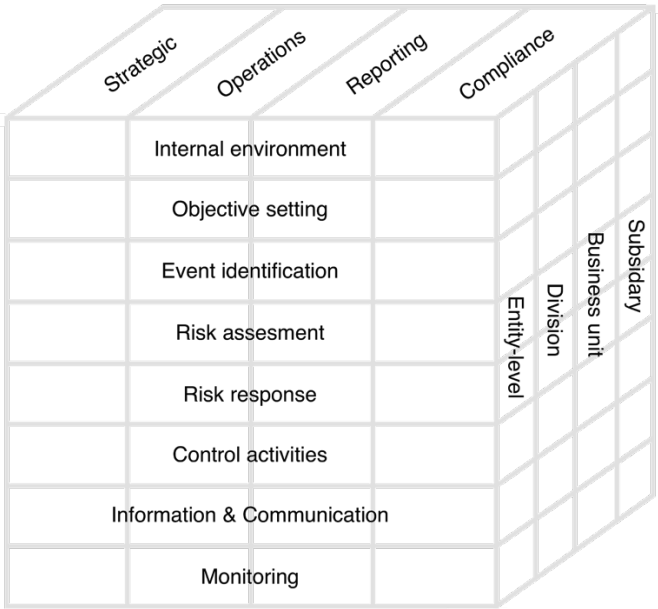
- **Granskning av källkoden till nya paket och uppdateringar** (Duan et al., 2020)**.** Att granska nya paket innebär analys av risk. Manuell granskning är givetvis mycket resursintensiv, Duan et al., 2020 har därför tagit fram ett automatiserat verktyg för granskning av kod. De föreslår att ett sådant verktyg bör inkluderas i beroendehanteringssystemet. Även Vaidya et al., 2019 föreslår att paket som väcker misstanke bör granskas noga, men till skillnad från Duan et al., 2020 understryker de att automatisk granskning är mycket svår.

- **Noggrannhet vid installation och uppdatering av paket: undvik "obskyra" eller övergivna paket och paket vars namn avsedda att förväxlas med andra** (Vaidya et al., 2019)**.** Författarna föreslår automatiska verktyg för att underlätta detta arbete.

2.1.4 Riskhantering, möjligheter och strategi

Risk och möjlighet är relaterade, där verksamheter tidigt i projekt bör utvärdera och definiera identifierade risker (Boehm & Bhuta, 2008). Vidare kan närvarande risker också påverka hur projektet ska genomföras, om en agil eller plandriven metod är att föredra (Boehm & Turner, 2003). Vilka risker som är närvarande kan alltså redan i planeringsfasen avgöra hur projekt utformas, men även typ av mål kan påverka riskhanteringsarbetet. I en studie av Islam et al. (2014) som genomförde en empirisk undersökning av GSRM (Goal driven Software Risk Model), det vill säga en måldriven mjukvaru-riskmodell, fann att GSRM kunde integreras i de flesta komplexa projekt för att underlätta riskhantering, och att det måldrivna perspektivet underlättade anställda att förstå och kommunicera om riskhanteringen. I GSRM identifieras mål och risker, samt länkar och kopplingar mellan dem och delprocesser för både projektet och riskhanteringsarbetet för att skapa en taxonomi som kan återanvändas i framtida projekt (Islam et al., 2014). En risk måste utvärderas mot mål som bakgrund, där vissa mål kan acceptera större risker. Det avgörs av en övergripande strategi.

Organisationen COSO skapade 2004 en modell för verksamheters riskhantering (denna är alltså inte framtagen specifikt för mjukvaruutveckling) som ger en översikt hur olika delar av verksamheten, dess strategier och riskhanteringsarbetet är kopplat.



Figur 2.5: COSO:s modell med fyra vertikala kolumner för målkategorier, åtta horisontella rader för delmoment i riskhantering och fyra kolumner i den tredje dimensionen för enheter i verksamheten (COSO, 2004).

Den här visualiseringen av COSO:s modell i figur 2.5 möjliggör en översikt där det är möjligt att se övergripande strategi för hela riskhanteringsarbetet eller gå in i mindre detalj per mål, delmoment, enhet eller möten mellan valda dimensioner (COSO, 2004). Den ger också perspektiv på hur riskhantering hänger ihop med mål och enheter där ingen av delarna existerar i ett vakuum, tvärtom påverkar de varandra.

2.2 Kärnaktiviteter

Ett möjligt problem med ovan beskrivna modeller är återkommande för modeller för riskhantering vid mjukvaruutveckling, ofta är de för generella för ett givet företags utvecklingsprocess, eller så kan de vara för specifika och inte fullt tillämpbara. Detta hävdar Boehm (Boehm, 2010) som kan ge ett perspektiv på hur modeller för och forskning om riskhantering anpassats över tid då mjukvaruutvecklingen förändrats, eftersom nya metoder och processer kräver ny forskning och nya modeller. Exempelvis tog Boehm fram en modell 2003 för att genom ett riskperspektiv bedöma ifall utvecklingen bör genomföras agilt eller plandrivet, 2008 publicerade han en modell som balanserar risker och möjligheter vid komponentbaserad mjukvaruutveckling och 2018 genomförde han en studie om hur defekter i OSS påverkar mjukvarukvalitet (Boehm & Turner, 2003; Boehm & Bhuta, 2008; Chen et al., 2018). Dessa exempel synliggör hur Boehms forskning blivit mer specifik (om än enbart i dessa fall) för att erbjuda lösningar till eller undersöka problem vid särskilda metoder och miljöer för mjukvaruutveckling.

2.2.1 Jämförelse av modeller och förslag

Det finns dock inga modeller för riskhantering vid utveckling med beroendehanteringssystem tillgängliga för denna undersökning, där krävs alltså nya. Istället för att försöka ta fram en ny modell ämnar denna sektion istället att identifiera återkommande aktiviteter eller delmoment från modeller och lyfta föreslagna återkommande metoder för att skapa en övergripande bild över vilka aktiviteter som skulle kunna vara hjälpsamma i riskhanteringen i denna nya miljö. Ett liknande grepp tar Bazaz et al. (2012) i en jämförande undersökning av modeller för bedömning av risker, där de finner att tre av de fyra undersökta modellerna innehåller identifiering av risk, samt att tre av dem innehåller prioritering av risker.

Tabell 2.2: Jämförelse av modeller för riskhantering. ✓ betecknar att metoden/aktiviteten finns i modellen, * betecknar att modellen medger att aktiviteten behöver inkluderas. Observera att Boehms spiralmodell är en modell för hela mjukvaruutvecklingsprocessen och inte enbart riskhantering

	Boehms spiral-modell	**ISO/IEC 16085**	**GSRM**
Koppling till övergripande strategi	✓	✓	✓

Planering av risk-arbete	✓	✓	✓
Identifiering av risk	✓	✓	✓
Analys av risk	✓	✓	✓
Prioritering av risker	✓	✓	✓
Arkiv av risker		✓	✓
Övervakning av risk	*	✓	✓
Behandling av risk	✓	✓	✓
Utvärdering av riskhantering	✓	✓	
Cyklisk	✓	✓	
Tilldela ansvar			✓
Höj kompetens-nivå	✓	✓	✓
Identifiera mål			✓

Genom tabell 2.2 synliggörs att vissa aktiviteter i riskhantering är återkommande över flera modeller, men också att ingen modell är heltäckande, framför allt när det gäller områden som är specifika för ett projekt och dess miljö. Det finns också fall där det är svårt att definitivt avgöra om en modell stödjer en given aktivitet eller ej, exempelvis riskprofil, där tre av fyra modeller klassificerats som att inte innehålla denna då de inte explicit uttrycker varken termen riskprofil eller uttrycker att status av risk ska uppdateras och arkiveras kronologiskt, men andra delaktiviteter kan återfinnas som att avgöra hur villigt ett projekt är att utsätta sig för vissa risker eller föra register för dem. Därför används här istället termen “Riskarkiv” för att fånga både riskprofil från ISO/IEC 16085 och taxonomin från GSRM.

Gemensamt för de jämförda modellerna är att arbeta utifrån en strategi som går i linje med verksamhetens övriga strategier och mål, för att sedan definiera metoder för varje aktivitet i riskhanteringen som passar bäst i sin kontext. I det aktiva arbetet är identifiering, analys, prioritering (analys och prioritering kan genom vissa metoder vara en aktivitet) och övervakning centrala aktiviteter, som med hjälp av strategi och tidigare planering ger en handlingsplan för hur ett utfall av en risk ska hanteras vilket informerar riskbehandlingen.

Vi gör också en jämförelse av förslag för riskhantering gällande tredjepartskomponenter för att identifiera specifika åtgärder i tabell 2.3. Dessa strategier kan ses som underkategorier till bland annat analys, övervakning och behandling.

Tabell 2.3: Jämförelse av förslag för hur organisationer bör arbeta med riskhantering vid utveckling med tredjepartskomponenter.

	Hauge et al. (2010)	**Duan et al., 2020**	**Vaidya et al., (2019)**
Använd kända, säkra komponenter	✓		✓ (undvik obskyra komponenter)
Övervaka antagna komponenter och deras källa/ community	✓	✓	✓
Installera uppdateringar skyndsamt		✓	
Dedikera personal med ansvar för analys och övervakning av komponenter	✓		
Använd privata paketregister		✓	
Granskning av källkod		✓	✓ (vid misstanke)

Strategierna i tabell 2.3 ser vi som underkategorier till några av aktiviteterna i tabell 2.2. Granskning av källkod är en typ av analys. Använd kända, säkra komponenter kräver både analys och planering, dedikering av personal är en sorts tilldelning av ansvar som också kan ingå under planering. Det finns två strategier här gällande övervakning, och användning av privata paketregister ser vi som en typ av behandling.

Nedan beskrivs de aktiviteter som med hjälp av tabell 2.2 och 2.3 identifierats som både återkommande kärnaktiviteter och relevanta för vår undersökning.

2.2.2 Övergripande strategi

Samtliga modeller föreskriver vikten av att använda en större och övergripande strategi. Det fungerar inte att enbart implementera ett riskhanteringsarbete med en egen strategi, utan den strategin måste fungera tillsammans med resten av verksamheten och dess mål. Exempelvis är GSRM en modell som består av ett flertal olika strategier och är samtidigt starkt anknuten till ett företags övergripande strategi genom mål (Islam et al., 2014). I spiralmodellen är riskhanteringsarbetet bara en delmängd av en betydligt större strategi för mjukvaruutvecklingen (Boehm, 1988). Risk i mjukvaruutveckling består av tre dimensioner - teknisk, organisatorisk och omvärld, där organisationens struktur, kultur och strategi samt faktorer utanför den alla bör tas hänsyn till för att skapa en god plan för riskhantering (Sherer, 1995).

2.2.3 Planering av riskhanteringsarbetet

Enligt ISO/IEC 16085 innehåller planering av riskhanteringsarbetet för mjukvaruutveckling delprocesserna etablering av policys för riskhantering, etablera riskhanteringsprocessen, tilldelning av ansvar och etablera utvärdering av riskhanteringen, samt att dessa strategier skall fungera väl tillsammans med verksamhetens övergripande riskhantering och strategi (ISO, IEC & IEEE, 2004). Boehm skriver samtidigt att vilken typ av risker som finns närvarande kombinerat med projektets utformning kan påverka inte bara planeringen av riskhanteringen, utan avgöra hur hela projektet bör se ut (Boehm & Turner, 2003).

a) Riskhantering implementeras, administreras, och stöttat av ledning och personal

b) Löpande engagemang i riskhantering från intressenter skapas och upprätthålls

c) Riskhanteringsprocessen koordineras mellan intressenter

d) Personal delaktig i riskhanteringen utbildas

e) Projektets riskprofil skickas ut till och blir granskad av intressenter, frekvensen av detta bestäms

f) Resurser för riskbehandling tillgängliggörs

Figur 2.6: Delmoment i etablering av policys för riskhantering enligt ISO/IEC 16085 (ISO, IEC & IEEE, 2004).

Figur 2.6 visar hur delmomentet etablering av policys i planeringen av riskhanteringsarbetet i ISO/IEC 16085 består av delprocesser, som i sin tur innehåller flera aktiviteter. Detta demonstrerar hur det oftast går att gå in i djupare detalj och granularitet i samtliga delmoment, samt att planeringen är betydligt mer utförlig än vad som kan tas upp i denna uppsats.

2.2.4 Identifiering av risk

Identifiering av risk är en ständigt pågående process, och med rätt strategi kan verksamheter bli mer proaktiva i sin riskhantering och utnyttja upptäckta möjligheter (Mishra et al., 2019). Det finns olika tillvägagångssätt för att identifiera risker, vanliga metoder är brainstorming, intervjuer, expertutlåtanden, antagande analys och grundorsaksanalys innan riskerna klassificeras exempelvis beroende på källan, på vilken fas de förväntas kunna uppstå eller på hur viktig risken upplevs vara (Hussain, 2017).

2.2.5 Analys av risker

I ISO/IEC 16085 bör ett beslut tas om vilken typ av riskanalys som ska utföras tidigt i planeringsstadiet där det finns två kategorier, kvalitativ och kvantitativ riskanalys (ISO, IEC & IEEE, 2004). Kvalitativa analyser använder matriser för att kartlägga risker beroende på sannolikhet och konsekvens av risken, kvantitativa metoder finns i många olika former men fungerar oftast genom att beräkna förväntade värden och en risks påverkan på dem (Hussain, 2017). Båda metoderna tillåter en kartläggning av risker som sedan ligger till grund för prioritering.

2.2.6 Prioritering av risker

Från analysen finns nu ett underlag för att prioritera mellan risker, där högt prioriterade risker övervakas mer noggrant och får fler resurser dedikerade (ISO, IEC & IEEE, 2004). Men framförallt för kvantitativ analys finns mängder av olika metoder som försöker lösa problemet med att korrekt kvantifiera osäkra omständigheter, där sitter ofta själva analysarbetet ihop med prioriteringen där de värden beräkningarna ger direkt resulterar i en prioriterad lista (Tang et al., 2018).

2.2.7 Riskarkiv

Termen riskarkiv används här för att innefatta både riskprofil och GSRM:s taxonomi. Riskprofilen fanns enbart i en av de jämförda modellerna, men delar av den återfanns även utanför ISO/IEC 16085. En riskprofil är ett kronologiskt ordnat register som håller samtliga versioner av information om risker vilket bland annat kan inkludera beskrivning, orsaker, sannolikhet, konsekvenser, värderingar, tillit till dessa värderingar, behandlingar etc., det finns även projektriskprofil, som håller samtliga riskprofiler och även kontexten för projektet och kan inkludera hur villigt projektet är att utsätta sig för vissa risker (ISO, IEC & IEEE, 2004). Riskprofilen i ISO/IEC 16085 överlappar till viss del med GSRM:s taxonomi, vilken också kan verka som ett register för information om risker, men även håller information om mål och hur risker och mål är kopplade till olika delar av ett projekt eller verksamheten och hur risker ska hanteras (Islam et al., 2014).

2.2.8 Övervakning av risk

Övervakning eller monitorering av risk innebär att kontinuerligt bevaka risker genom projektets livscykel (Islam et al., 2014). Detta kan ske via en mängd metoder beroende på typ av risk och omständigheter, exempelvis kan det göras genom att spåra status på system eller

bedöma riskens närvaro beroende på fas eller prioritet för att avgöra om riskens status ändras (Prasad et al., 2015; ISO, IEC & IEEE, 2004).

2.2.9 Behandling av risk

Hur ett utvecklingsprojekt bör bemöta en given risk och behandla den beror på kontexten, där flera typer av strategier kan appliceras, och hänger ihop med övervakningen då en risks behandling ändras först efter att riskens status ändrats (Medhioub et al., 2017). Behandlingen sker sedan enligt den plan som tagits fram för risken och innehåller behandlingsaktiviteter och metoder, schema, resurser, ansvarsfördelning etc., förutsatt att behandlingens kostnad inte överstiger riskens konsekvenser (ISO, IEC & IEEE., 2004). Behandling av risk är alltså vad som ofta kallas mitigation eller treatment, och sker inte bara efter att en risk utfallit utan innefattar också hur verksamheter arbetar förebyggande för att minska sannolikheten och konsekvensen av en risk.

2.2.10 NPM-specifika åtgärder och best practices

Hauge et al. (2010) och Duan et al. (2020) beskriver ett flertal strategier för hur risker vid utveckling med tredjepartskomponenter kan hanteras. Duan et al. (2020) beskriver åtgärder som är specifika för NPM, vilka i stor utsträckning är tydligare NPM-specifika definitioner av förslagen från Hauge et al. (2010) och gäller vikten av att ha ett privat paketregister, utföra regelbunden kontroll av säkerhetsråd och sedan skyndsamt installera säkerhetsuppdateringar, samt att nya paket och uppdateringar bör granskas. Hauge et al. (2010) föreslår även att dedikera personal med ansvar för analys och övervakning av komponenter.

2.3 Riskhantering i agil mjukvaruutveckling

Denna undersökning ämnar inte undersöka hur riskhantering fungerar vid just agil utveckling, men då en studie fann att 97% av undersökta mjukvaruutvecklingsföretag hade agila processer (Hoda et al., 2018) samtidigt som merparten av verksamheterna i vår undersökning använder agila utvecklingsmetoder fann vi det lämpligt att kort lyfta detta eftersom det kan påverka ett företags riskhantering. Det är en problematik som var viktig för oss att vara medveten om vid genomförandet av undersökningen, samt för läsaren för att kunna förstå den kontext företagen verkar i.

2.3.1 Informell riskhantering

Implementation av riskhantering kan vara en utmaning vid agil utveckling, då de modellerna som presenteras hittills i kapitel 2.2 inte nödvändigtvis är direkt implementerbara. De identifierade kärnaktiviteterna bör dock kunna finnas i någon form, men ofta saknas uttalade riktlinjer helt. Hammad et al. (2019) genomförde en empirisk undersökning om hur verksamheter arbetar med riskhantering i agila projekt (vars natur med mål om snabb utveckling är direkt motsägelsefullt till god riskhantering), där de fann att riskhanteringsstrategier fanns i viss utsträckning, men oftast inte formaliserat. Resultaten visade att endast 17% av svarande företag hade dedikerade roller för riskhantering, 52% hade ingen kommunikation om riskhantering alls i sina projekt och 56% varken övervakade eller kontrollerade risker (Hammad et al., 2019). Agil utveckling har enligt undersökningen betydande brister i sin riskhantering, där de

mest centrala aktiviteterna kan saknas. Men agila utvecklingsmetoder är riskdrivna i den mån att ständig återkoppling från kund finns, risker ignoreras inte helt men riskhanteringen sker främst enligt en implicit process där en explicit struktur hade hjälpt att fånga många risker som passerar obemärkt och kan påverka framgången av projekt (Hammad & Inayat, 2018). Dock fann Siddique & Hussein (2014) i en undersökning av riskhantering i agila projekt för mjukvaruutveckling i Norge att de flesta svarande hanterade risker på samma sätt som i utveckling med vattenfallsmodellen, där exempelvis en riskmatris skapas. Detta görs med risker identifierade genom brainstorming tillsammans med kund, som sedan analyseras och för de med hög prioritet görs behandlingsplaner, prioriterade risker listas, övervakas och behandlas, där kompletterande strategier är att ha korta sprintar och utföra SWOT-analyser (Siddique & Hussein, 2014). Dessa fynd är i konflikt med varandra, men visar sammantaget att det finns möjligheter att förbättra och formalisera riskhantering i agila projekt. Det finns därför föreslagna modeller för hur riskhantering kan integreras med agila utvecklingsmetoder. Olika metoder för agil utveckling som SCRUM, Kanban, XP och DSDM, har olika styrkor och svagheter gällande riskhantering och exponering för risk (Agrawal et al., 2016). Det viktiga med dessa integrationsmodeller är att visa hur det går att införa riskhanteringsstrategier och de identifierade kärnaktiviteterna i ett projekt även om projektets agila struktur inte tillåter en omedelbar överföring av en traditionell modell.

Hammad och Inayat (2018) och Nyfjord och Kajko-Mattson (2008) presenterade två modeller för hur riskhantering kan integreras i agila processer vid mjukvaruutveckling, som i båda fall visades hjälpa organisationer hantera risker mer framgångsrikt. De två integrationsmodellerna tar två olika infallsvinklar då Hammad och Inayat (2018) främst passar in riskhanteringsaktiviteter i existerande moment i SCRUM, där Nyfjord och Kajko-Mattson (2008) istället skapar ett nytt ramverk ovanpå existerande organisation med nya delmoment och en helt ny avdelning, vilket är tänkt att kunna fungera till fler agila metodologier än en. Det finns alltså sätt att integrera ett sunt riskhanteringsarbete även i projekt vars agila struktur till synes borde försvåra riskhantering.

2.4 Litteratursammanfattning

Här presenteras i tabell 2.4 en sammanfattning av den teoribildande litteraturen.

Tabell 2.4: Litteratursammanfattning

Område	Beskrivning	Författare
Övergripande strategi	Strategin för verksamhetens riskhantering bör fungera väl ihop med dess övergripande strategi	Boehm (1988), ISO, IEC & IEEE (2004), Islam et al. (2014), COSO (2004)
Planering av riskhanteringsarbete	Planering och definition av aktiviteter och roller i riskhanteringsarbetet	Boehm (1988), ISO, IEC & IEEE (2004), Islam et al. (2014), Boehm & Turner (2003)
Identifiering av risk	Aktivitet där risker identifieras och klassificeras	Boehm (1988), ISO, IEC & IEEE (2004), Islam et al. (2014), Mishra et al. (2019), Hussain (2017), Bazaz et al., (2012)
Analys av risker	Aktivitet för utvärdering av risker, behandlingsplaner tas fram	Boehm (1988), ISO, IEC & IEEE (2004)), Islam et al. (2014), Hussain (2017), Bazaz et al. (2012)

Prioritering av risker	Risker prioriteras, ofta via riskvärden	Boehm (1988), ISO, IEC & IEEE (2004), Islam et al. (2014), Bazaz et al. (2012), Tang et al. (2018)
Riskarkiv	Arkiv av tidigare identifierade och bedömda risker	ISO, IEC & IEEE (2004), Islam et al. (2014)
Övervakning av risk	Kontinuerlig övervakning av riskers status, kan ske manuellt, enligt fas eller med hjälp av system	ISO, IEC & IEEE (2004), Islam et al. (2014), Prasad et al. (2015)
Behandling av risk	Hur risker bemöts både innan och efter utfall, enligt framtagen plan	Boehm (1988), ISO, IEC & IEEE (2004), Islam et al. (2014), Medhioub et al. (2017)
Använd kända, säkra komponenter	En typ utav proaktiv riskbehandling vilket innebär att enbart andvända etablerade, erkända, komponenter med bevisad framgång.	Hauge et al. (2010); Vaidya et al., (2019);
Övervaka antagna komponenter och deras källa/community	Löpande riskövervakning under arbetets gång	Hauge et al. (2010), Duan et al., (2020), Vaidya et al., (2019)
Privata paketregister	En form av riskbehandling via ett privat centraliserat paketregister	Duan et al. (2020)
Regelbunden kontroll av säkerhetsråd	En typ av övervakning, söker efter säkerhetsuppdateringar för inkluderade beroenden	Duan et al. (2020)
Skyndsam installation av uppdateringar	Att omgående installera säkerhetsuppdateringar	Duan et al. (2020)
Dedikera personal med ansvar för analys och övervakning av komponenter.	Tilldelning av ansvar och roller för riskövervakning.	Hauge et al. (2010)
Granskning av källkod /paketinnehåll.	En typ av analys av paket och beroenden	Duan et al. (2020); Vaidya et al., (2019)
Riskhantering i agil utveckling	Agil utveckling kan innebära försämrat riskhanteringsarbete som ofta sker implicit, men det är möjligt att implementera formaliserad riskhantering	Hoda et al. (2018), Hammad et al. (2019), Hammad & Inayat (2018), Siddique & Hussein (2014), Agrawal et al. (2016), Nyfjord & Kajko-Mattson (2008)

3 Metod

3.1 Litteratursökning

Oates (2006) beskriver en litteraturgenomgång som en process i sju steg: sökning, erhållande, bedömning, läsning, kritisk utvärdering och skrivande. Första steget, sökningen, inleds genom att sammanställa en lista med relevanta termer och nyckelord. Oates (2006) rekommenderar att man börjar med en fras, delar upp den i olika koncept och hittar alternativa termer för varje koncept. Vi valde att utgå ifrån uppsatsen titel: "Riskhantering vid mjukvaruutveckling med NPM", vilket resulterade i följande koncept och alternativa termer. Begreppen översattes till engelska då vi snabbt blev varse om att utbudet av relevant forskning på svenska var begränsat.

Tabell 3.1: Koncept och alternativa termer

Riskhantering	Mjukvaruutveckling	NPM
Risk management Risk assessment	Software development	Node package manager Software ecosystem Open source software Javascript

Vi inledde sedan sökningen genom att använda olika kombinationer av dessa termer i sökmotorerna LUBSearch och Google Scholar. Detta resulterade i ett antal artiklar som blev vår utgångspunkt i det fortsatta sökandet. Genom dessa artiklar upptäckte vi nya relevanta sökord, samt forskare som inriktat sig på riskhantering och beroendehanteringssystem. Dessa fynd blev sedan grund för nya sökningar.

Det andra steget i processen, "erhållande" utfördes parallellt med den inledande sökningen. Samtliga utav våra källor, med undantag boken av Oates (2006), fanns tillgängliga via internetbaserade databaser. Detta möjliggjorde att även bedömning och läsning i viss mån kunde ske parallellt med sökningen.

Vid bedömning av texter använde vi oss av bl.a. utav följande kriterier. Givetvis bedömdes också texternas kvalitet och relevans.

- Texten ska ha genomgått peer review.
- Texten ska vara publicerad i en respekterad publikation eller konferens.
- Upphovsmännen bör vara etablerade inom fältet
- Det är en fördel om texten citerats av andra forskare.

För att underlätta läsning, granskning och katalogisering skrevs texterna ut. Vid genomläsningen gjordes överstrykningar och anteckningar i marginalerna i syfte att synliggöra den information vi ansåg vara av intresse. Texter som klarade vår initiala granskning sparades i referenshanteringssystemet Zotero.

3.2 Val av metod för empirisk undersökning

För den empiriska undersökningen valde vi en kvalitativ metod, där respondenter intervjuades semistrukturerat med hjälp av en intervjuguide. Detta såg vi som det bästa alternativet av flera anledningar, bland annat för att de typer av svar vi sökte var mer utredande där vi ville skapa en djupare förståelse av vad som ligger bakom de val som gjorts av respondenter och även förstå hur deras arbete med riskhantering för NPM-paket påverkas av andra faktorer i verksamheten. Även om distinktionen mellan kvalitativ och kvantitativa metoder inte helt ligger i vilken typ av data som kan samlas in, brukar kvalitativa undersökningar ses som mer lämpade för att kunna generera denna typ av svar, där kvantitativa undersökningar är bättre för att ta fram numeriska data (Bryman, 2012). Vi önskade alltså kunna få detaljerade svar, där den semistrukturerade intervjun kan få fram detaljer genom följdfrågor, och tillåter ordningen av frågor och ämnen att ändras beroende på respondentens svar, samt kan ge respondenten möjlighet att självmant berätta om ämnen de tror kan vara av intresse för undersökningen (Oates, 2006). Enligt Oates (2006) är den semistrukturerade intervjun bra för undersökningar där syftet är "discovery" eller upptäckt snarare än att kontrollera och då lämpliga för mer djupgående undersökningar, men är sämre för att göra större generaliseringar eftersom frågorna och svaren inte kommer att överensstämma helt mellan intervjuer samt att metoden oftast innebär ett färre antal respondenter. Valet att utföra undersökningen med en semistrukturerad intervju tillät oss att få betydligt mer detaljerade svar än vad som skulle ha varit möjligt med en kvantitativ studie eller en strukturerad intervju, samtidigt som olika intervjuer blir mer lika och lättare att jämföra och återskapa än i en ostrukturerad intervju.

3.3 Val av respondenter

För att besvara forskningsfrågorna ville vi intervjua personer på företag som ägnar sig åt systemutveckling och som använder NPM-paket i den utvecklingen. Vi ville också ha svaranden från företag i olika storlekar och med olika förutsättningar, för att se om detta kunde påverka hur verksamheter arbetar med sin riskhantering. Vi hoppades kunna hitta respondenter som hade insikt i både hur företaget arbetar med riskhantering, och på detaljnivå kunde beskriva arbetet med NPM-paket och hur de förhåller sig till det. Vi skickade därför ut intervjuförfrågningar till flera olika företag och kunde boka intervju med respondenter från fem olika företag i olika storlekar. Här följer en kortare beskrivning av varje företag och respondent, notera att siffrorna aldrig är exakta för att anonymisera verksamheterna och respondenterna.

Företag 1, hädanefter kallat *E-handelskonsultbolaget*, arbetar med vad de kallar e-commerce, där de hjälper kunder med allt inom e-handel, men även tar fram system som ska finnas på plats i butiker. De finns med i en tjänsts hela livscykel, och arbetar mycket utifrån olika plattformar de själva tagit fram som en bas på vilken de sedan bygger individuella lösningar till kunden på. De har över 200 anställda, finns i flera länder och omsätter mer än SEK 250 miljoner. Respondenten var en full-stack-utvecklare.

Företag 2, hädanefter kallat *Produktutvecklingsbolaget*, arbetar med systemutveckling i två skilda syften, och då på två olika sätt. Främst konsulterar de större företag, där de har kunder över hela världen som inkluderar världsledande företag inom konsumentteknologi, och hjälper dem i innovationsprojekt med att ta fram designprototyper. Men de har också utvecklat ett eget verktyg som underlättar framtagandet av dessa prototyper, vilket de erbjuder och underhåller till vissa av deras kunder. De har under 50 anställda och omsätter under SEK 50

miljoner. Respondenten var en medgrundare av företaget som arbetar både med kundprojekt och med utveckling av prototyp-verktyget.

Företag 3, hädanefter kallat *AI-konsultbolaget*, är ett nytt konsultbolag, som funnits i ett och ett halvt år och fokuserar på data engineering, data science och machine learning. De konsulterar främst åt startups, och arbetar både med att ta fram prototyper och färdiga system. De har under 10 anställda och omsatte mindre än SEK 10 miljoner förra året. Respondenten beskriver sig själv som tekniskt ansvarig och partner.

Företag 4, hädanefter kallat *Vårdbolaget*, är ett internationellt vårdbolag, som har en specifik typ av kliniker i flera världsdelar. Klinikerna är företagets kärnverksamhet, och de har en intern IT-avdelning som utvecklar system som ska användas på dessa kliniker. De har över 10 000 anställda och omsätter mer än SEK 5 miljarder. Respondenten är konsult inom frontendutveckling men har arbetat med företaget i ett antal år.

Företag 5, hädanefter kallat *Säkerhetsbolaget*, är ett nytt företag som säljer en tjänst framtagen för att hjälpa kunder med en del av de problem arbete med beroendehanteringssystem som NPM kan medföra, bland annat genom att automatisera kontroll av säkerhetsuppdateringar och licenser. De har över 10 anställda och omsatte mindre än SEK 1 miljon förra året. De två respondenterna är en junior utvecklare samt en medgrundare och CTO.

Ett problem vid valet av respondenter var att hitta en person som hade mycket god insikt i både riskhanteringsarbetet och hur verksamheten arbetar med NPM. Vi valde här att prioritera personer som arbetade direkt med NPM. Då vissa företag saknade formaliserad riskhantering helt ville vi se till att intervjuerna kunde bli så lika som möjligt genom att intervjua respondenter med liknande roller på olika företag, där alla direkt arbetar med NPM. Detta innebar att respondenter ibland hade svårt att svara på exakt hur riskhanteringen fungerade i stort på företaget.

3.4 Intervjuguide

Vi tog fram en intervjuguide som var uppdelad i tre delar. Först ett par enklare faktafrågor, sedan frågor utifrån scenarion, och till sist återkopplande frågor.

Inledande frågor	• Önskar du anonymiseras med pseudonym i uppsatsen? Skall företaget anonymiseras? • Hur skulle du beskriva företagets verksamhet? o Är systemutveckling en kärnverksamhet? • Vilken roll har du på företaget? • Hur stort är företaget? • Var har ni för kunder? • Arbetar ni agilt? • Använder ni NPM-paket i er utveckling?

Scenario	• Företaget ska dra igång ett nytt utvecklingsprojekt.
	  o Hur ser riskhanteringsarbetet ut i detta inledande skede?
	  ▪ Vilka aktiviteter finns - Planering, identifiering, analys, prioritering, övervakning?
	  ▪ Tilldelas ansvar inom riskhantering?
	  ▪ Kan identifierade risker påverka hur projektet planeras?
	  o Används någon form av arkiv eller lista med tidigare identifierade risker?
	  ▪ Uppdateras denna löpande?

	○ Fattas beslut kring arkitektur och val av tredjepartskomponenter i detta skede? • En utvecklare har tilldelats en uppgift och ser en möjlighet att spara tid och arbete genom att inkludera ett externt beroende istället för att "återuppfinna hjulet". ○ Får utvecklaren lov att göra detta? ▪ Vem fattar beslutet? ▪ På vilka grunder? ○ Sker någon granskning av paketet? ▪ Enligt vilka kriterier? ▪ Vem utför den? ○ Vad händer om beroendet godkänns? ▪ Upprättas dokumentation (BoM)? ▪ Hur säkerställs fortsatt tillgång? • Lagras kopia i lokalt paketregister? • Systemet befinner sig nu i skarp drift. ○ Hur ser det löpande beroendeehanteringsarbetet ut? ○ Finns det en roll med ansvar för att övervaka uppdateringar och rapporter om sårbarheter? ○ Hur ser det övriga, mer övergripande, riskhanteringsarbetet ut under projektets gång? ▪ Sker kommunikation om risker? ▪ Övervakas de? • En uppdatering till beroendet publiceras. ○ Inkluderas det uppdaterade beroendet i projektet? ▪ Hur påverkas beslutet av uppdateringens syfte (säkerhetsuppdatering eller inte)? ▪ Hur sker detta? Används semantisk versionshantering? ○ Granskas det uppdaterade paketet på nytt? ▪ Hur lång tid tar det?
Återkoppling	• Strategi: NPM - Riskhantering - Total strategi för hela verksamheten • Tidsdimension: Hur har arbetet med NPM förändrats över tid? • Hur villig är man att ta en risk? Hur stor är chansen vs. hur stor är konsekvensen? Är vissa "större risker" ändå värda att ta? • Hur ser ni på fördelarna med att använda NPM? Vad hade konsekvenserna blivit om ni inte gjorde det? Vad gjorde ni innan?

Vi valde att inleda intervjun med dessa enklare frågor, dels för att snabbt fastställa och kontrollera att vår uppfattning av respondenten och företaget är korrekt, samt använda denna information för att kontextualisera svaren (Bryman, 2012). Men även för att det är bra att börja med frågor respondenten har lätt att svara på (Oates, 2006). I själva huvuddelen av intervjun förespråkar Oates (2006) användning av s.k. öppna frågor, dvs frågor som börjar med "vad?", "hur?" eller "varför?". Denna typ av frågor låter respondenten svara mer fritt, snarare än att bara bekräfta eller motsäga ett påstående. Hon föreslår också att dessa frågor placeras i en kontext, t.ex. med olika hjälpmedel eller genom att beskriva ett scenario. Vi valde därför att konstruera ett scenario som beskriver olika steg i ett utvecklingsprojekt. Ett sådant scenario underlättade såväl intervjun som den efterföljande analysen, då det blev enklare att jämföra olika svar trots företagens varierande strategier och arbetssätt. Samtidigt tillät dessa scenarier dels förberedda följdfrågor, men också spontana följdfrågor efterhand vi som intervjuare skapade en bättre förståelse för hur respondenten arbetar (Bryman, 2012).

Till sist innehåller intervjuguiden ett par bredare frågor, som tar ett större perspektiv än ett givet scenario. De tänktes användas dels för att kunna få en bättre förståelse för respondentens resonemang om val av arbetssätt, men också för att kunna generera en diskussion om andra större ämnen, och utforska om det fanns en koppling mellan företagets övergripande strategi och det konkreta arbetet med NPM.

Ett problem var att formulera frågor för att få fram information om hur respondenterna arbetar enligt teorin. Dels beskriver Bryman (2012) hur en strategi för att undvika ledande frågor är att inte ställa frågor direkt gällande det undersökningen vill ha svar på, där det kan bli problematiskt att exempelvis ställa frågor som alltid är direkt anknutna till teorin. Vi inkluderade ändå frågor om varje aktivitet i riskhantering, för att kunna ställa följdfrågor ifall respondenten hade kunskap om detta. Vi fann det vara en svår balans att inte göra det tydligt för respondenten vilka typer av svar vi var intresserade av, samtidigt som vi var tvungna att locka fram de svaren. Annars ämnade vi att undersöka hur de arbetade med risker gällande NPM, för att sedan se om det arbetssätt var implicit konsekvent med riskhantering. Det är viktigt att ställa frågor som är kopplade till teorin som används, men det är samtidigt viktigt att formulera frågor som är relevanta för respondenterna och med ett språk de kan förstå (Bryman, 2012).

3.5 Genomförande av intervju

Samtliga intervjuer genomfördes online, fyra via videokonferens och en via ljudkonferens, då den rådande Covid-19-pandemin försvårade intervjuer på plats. Vid samtliga intervjuer deltog båda författarna.

Tabell 3.2: Översikt av undersökningens intervjuer

	E-handelskonsultbolaget	**Produktutvecklingsbolaget**	**AI-konsultbolaget**	**Vårdbolaget**	**Säkerhetsbolaget**
Respondent	Full-stack-utvecklare	Medgrundare / utvecklare	Partner / Tekniskt ansvarig	Front-end-konsult	Junior Developer & CTO/Medgrundare
Datum	29/4–2020	30/4–2020	6/5–2020	6/5–2020	11/5–2020
Längd	43	45	33	35	36
Plattform	Videosamtal, Microsoft Teams	Videosamtal, Google Meets	Videosamtal, Zoom	Ljudsamtal, Slack	Videosamtal & ljudsamtal ,Google Meets
Utrustning för inspelning	Microsoft Teams	Google Meets, Quicktime	Zoom, Quicktime	Quicktime, Iphone	Quicktime, Iphone
Önskar vara anonym	✓	×	✓	✓	✓
Appendix	A	B	C	D	E

Inför intervjun förberedde vi oss dels genom att vara väl bekanta med intervjuguiden, och med vad vi ville få fram för typ av information från respondenten. Vi förberedde oss också genom att skapa en grundläggande uppfattning av verksamheten respondenten arbetade i, för att ha en tydligare kontext till svaren (Bryman, 2012).

Vi använde oss av intervjuguiden, där vi lät tidigare svar informera följdfrågor, eller samtalet ibland kom att i förväg besvara senare frågor. Ett möjligt problem var att de öppna frågorna ibland resulterade i svar som inte var fullt så detaljerade som vi önskade, därefter ställde vi preciserande följdfrågor som kan ha varit ledande. Både Bryman (2012) och Oates (2006) beskriver vikten av att undvika ledande frågor, då de kan påverka svaren och därmed undersökningens tillförlitlighet. För oss gällde det att få en hög validitet, där vi bedömde vikten av att försäkra oss om att frågan uppfattats korrekt och få ett precist svar som större än hotet av ledande frågor.

Oates (2006) pekar på ett antal problem med internetbaserade intervjuer, bland annat att mycket information och förståelse kan gå förlorad om man inte har möjlighet att höra varandra eller tolka varandras kroppsspråk. På grund av det rådande läget med en pågående pandemi ansåg vi oss inte ha något val. Samtliga intervjuade företag har infört distansarbete för att minska risken för smittspridning. Mot bakgrund av detta vore det olämpligt för oss att besöka dem eller deras anställda, då ett sådant förfarande hade inneburit ökad risk för såväl oss som dem. Vi har därför försökt efterlikna en fysisk intervju så nära som möjligt, bland annat genom att använda oss av videosamtal.

Inspelning av intervjun för transkription skedde oftast via plattformens inbyggda funktion, i senare intervjuer användes även Quicktime för att spela in ljud lokalt på en av författarnas datorer som en backup. I intervjun med Vårdbolaget möttes vi av tekniska problem, som innebar att vi var tvungna att snabbt byta plattform till Slack och utföra intervjun enbart med ett röstsamtal. Slack har inte någon inbyggd inspelningsfunktion, och därför spelades samtalet in med Quicktime samt Iphones medföljande inspelningsfunktion. I intervjun med Säkerhetsbolaget pratade vi med två respondenter på företaget, varav den ena var något sen till mötet på Google Meets och inte använde video. Precis som Oates (2006) säger upplevde vi att i de fall videosamtal inte användes förlorade vi ytterligare viss förståelse som annars kan uppfattas via kroppsspråk och mimik, och samtalen fick ett sämre flyt med fler avbrytningar då det var svårare att förstå när en part hade pratat klart.

3.6 Etik

Oates (2006) beskriver ett flertal kriterier för hur en undersökning bör genomföras etiskt:

- Rättigheten att inte delta
- Rättigheten att dra sig ur
- Rättigheten att ge informerat samtycke
- Rättigheten att vara anonym
- Rätt till konfidentialitet

Punkterna tangeras också i viss mån av Bryman (2012) som skriver följande:

- Huruvida det finns skada för deltagare;
- Huruvida det finns en brist av informerat samtycke;
- Huruvida det finns ett intrång i den personliga integriteten;
- Huruvida bedrägeri eller villfarelse är inblandat.

Vi beaktade hur Oates (2006) och Bryman (2012) beskrev att forskning, undersökningar och intervjuer bör utföras etiskt i vårt genomförande. Vid den initiala kontakten via mail var vi

tydliga med att syftet för intervjun var att undersöka hur företaget arbetade med riskhantering relaterat till deras arbete med NPM, där resultaten sedan skulle komma att publiceras i en C-uppsats vid Lunds Universitet, samt att författarna var två studenter som läste systemvetenskap. Vi informerade dem även om att intervjun beräknades ta ca 30–45 minuter och kunde genomföras på distans, samt att det var möjligt att vara anonym om så önskades. Vid de tillfällen företag tackade nej men ändå hade varit intressanta för oss att intervjua, accepterade vi det svaret då alla företag och respondenter har rätt att inte delta. Vi valde sedan att genomföra intervjuerna online på de plattformar respondenterna var mest bekväma med och använde i sin vardag.

Intervjuerna inleddes med att bekräfta att respondenten ville delta, sedan tacka respondenten för deras medverkan. Därefter förklarade vi att intervjun skulle spelas in, med syfte att transkribera intervjun samt att inspelningen kunde fungera som protokoll ifall en dispyt om vad som faktiskt sagts skulle uppstå, men att ingen annan skulle få ta del av inspelningen annars. Respondenterna informerades även om att de har rätt att dra tillbaka sitt samtycke innan publicering, och uppmanades att höra av sig till oss ifall frågor uppkom. Efter att detta godkänts av respondenten startades inspelningen. Den första frågan berörde ifall respondenten och företaget skulle anonymiseras. I ett fall svarade en respondent att hen och företaget inte behövde anonymiseras, men ändrade sig efter intervjun avslutats och bad om att bli anonym. Vid intervjuns slut förtydligade vi att inspelningen nu är avslutad, och tackade ännu en gång respondenten för dess medverkan.

Då vår undersökning behandlar frågor om IT-säkerhet, och kan komma att exponera svagheter hos de tillfrågade organisationerna anser vi det lämpligt att anonymisera samtliga respondenter, även i fall då de inte begärt detta. Det finns en risk att information som framkommer i undersökningen hade kunnat användas för att rikta attacker mot de medverkande företagen, samt att svar som inte ligger i linje med s.k. best practices kan skada deras anseende.

Intervjuerna spelades in och inspelningarna lagrades tillfälligt på författarnas datorer i väntan på att transkriberas, efter transkribering flyttades inspelningarna till molntjänsten Google Drive, som är skyddad med tvåfaktorsautentisering, och därmed får anses vara mycket säker. Inspelningarna kommer endast att lämnas ut ifall det blir nödvändigt att bevisa att undersökningen genomförts på ett korrekt sätt. Transkriptionerna är anonymiserade och kan inte användas för att identifiera respondenterna eller organisationerna de tillhör. Exempelvis valde vi att förutom dölja namn på respondenten och dess företag, även dölja namn på kunder eller samarbetande företag, samt dölja exakta siffror vilka annars kan användas för att identifiera respondentens företag, som storlek och omsättning.

3.7 Transkription och analys

Även om det är viktigt att vara detaljerad i transkriptionen, valde vi att bortse från stamningar och ifyllnadsljud som “eh” eller “öhm”, och inkluderade heller inte information om pauser eller tonalitet då vi inte ansåg att det var viktigt för att tolka svaren (Oates, 2006). Samtidigt skriver Bryman (2012) att det är centralt att ha med denna typen av information i transkriptionen, men vi bedömde att det gäller en annan typ av forskning som ämnar att utforska respondenten själv, och dennes känslor. Vi uteslöt dock inte denna typ av information helt, då vi exempelvis inkluderade skratt. Vi valde även att numrera raderna i transkripten, för att förenkla hänvisningar för både läsaren och vår egen analys.

För analys av den insamlade datan valde vi en kvalitativ analys för att hitta mönster och teman i det empiriska materialet (Oates, 2006). Segment av transkriptionen kategoriserades enligt teorier från litteraturen, vilket innebar vad Oates (2006) kallar deductive approach där vi var vaksamma för att inte mista upptäckter grundade i själva datan. Segmenten placerades i olika tabeller per kategori, där vi sedan sökte teman och likheter eller olikheter mellan olika segment i varje kategori. Därefter söktes även kopplingar mellan segment och teman i olika kategorier.

Kategorierna som användes var de som presenterades i kapitel 2.4.

- Övergripande strategi
- Planering av riskhanteringsarbetet
- Identifiering av risk
- Analys av risk
- Prioritering av risker
- Riskarkiv
- Övervakning av risk
- Behandling av risk
- Privata paketregister
- Regelbunden kontroll av säkerhetsråd
- Skyndsam installation av uppdateringar
- Granskning av nya paket och uppdateringar

3.8 Validitet och reliabilitet

Oates (2006) beskriver att en fördel med semistrukturerade intervjuer är att de kan medföra en högre validitet än i strukturerade intervjuer eller kvantitativa undersökningar. Detta kan nås dels genom att få mer specificerande svar, och att kontrollera att frågor och svar uppfattas korrekt. Oates (2006) nämner även ett par nackdelar med den semistrukturerade intervjun. Gällande validitet utgår studien från svaren som kanske inte alltid är korrekta, även om Oates (2006) samtidigt skriver att intervjuaren kan vara i en bra position för att bedöma just korrektheten i svaren. Den kanske främsta nackdelen enligt Oates (2006) är att intervjuer innebär en svagare reliabilitet, där intervjuerna är svåra att replikera exakt. Den semistrukturerade intervjun är här något bättre än den ostrukturerade, men problem om hur en enskild intervjus utformning beror på dess unika kontext kvarstår och påverkar reliabiliteten negativt. Vi var medvetna om denna problematik vid valet av semistrukturerad intervju som undersökningsmetod, vilken vi fann mest lämpad för denna studie som är av explorativ art. Vi bedömde att en semistrukturerad intervju skulle tillåta oss att få fram den nivå av detalj som krävdes, och att följdfrågor skulle vara nödvändiga för det samtidigt som det skulle kunna innebära en högre validitet genom följdfrågor. Ett annat problem som då uppstod är konflikten med ledande frågor kontra preciserade svar. Ledande frågor är ett hot mot validiteten, men vi upplevde att de ibland var nödvändiga för att kommunicera till respondenten precis vad vi menade i följdfrågor. Vi använde först öppna frågor, och vid behov sedan en följdfråga för att få ett precist svar eller försäkra oss om att vi uppfattat svaret korrekt. Vid analys av transskript var vi sedan försiktiga med att inte dra för stora slutsatser från svar, utan ämnade att enbart utgå från det som explicit sagts i intervjun. Ett annat hot mot validiteten var att intervjuerna var tvungna att ske på distans, där det då är möjligt att förlora förståelse. Vi hade helst genomfört intervjuerna på plats, men det var dessvärre inte möjligt på grund av Covid-19-pandemin.

4 Empiriskt resultat och analys

I detta kapitel presenterar och diskuterar vi resultaten från vår undersökning. I analysen identifierade vi 12 kategorier främst grundade i teorin, och här visar vi först de mönster som upptäcktes inom varje kategori innan vi beskriver samband mellan kategorierna. Detta görs först genom en enklare tabell för att visa grundläggande skillnader och likheter mellan respondenternas företag per kategori, och sedan beskriva dessa mönster.

Vi kommer i detta kapitel hänvisa till de transkriberade intervjuerna i appendix A-E, genom att ange radnummer som "A12", där "A" betecknar vilken bilaga det rör sig om, och "12" vilket radnummer i det transkriptet vi hänvisar till. Vi kommer också använda oss av tabeller för att presentera resultat, där "✓" betecknar att en aktivitet finns i företaget, och "×" innebär att den saknas.

För att ge en snabb inblick i skillnaden och likheterna mellan företagens riskhantering placerar vi in dem i en tabell som innehåller samma faktorer som användes för att jämföra modeller i litteraturgenomgången, innan resultaten från analysen presenteras mer detaljerat.

Tabell 4.1: En bred och förenklad översikt av de undersökta företagens riskhanteringsarbete enligt kärnaktiviteterna funna från jämförelsen i tabell 2.2, kapitel 2.2.1

	E-handelskonsultbolaget	**Produktutvecklingsbolaget**	**AI-konsultbolaget**	**Vårdbolaget**	**Säkerhetsbolaget**
Koppling till övergripande strategi	✓	✓	✓	✓	✓
Planering av riskarbete	✓	×	×	✓	×
Identifiering av risk	✓	×	×	✓	✓
Analys av risk	✓	×	×	✓	✓
Prioritering av risker	✓	×	×	✓	×
Arkiv av risker	✓	×	×	✓	×
Övervakning av risk	✓	Delvis, automatiserad	Delvis, automatiserad	✓	Delvis, automatiserad

Behandling av risk	✓	×	✓	✓	✓

Tabell 4.1 erbjuder en snabb överblick som visar att E-handelskonsultbolaget och Vårdbolaget är ganska lika i sitt arbete, Produktutvecklingsbolaget och AI-konsultbolaget är ganska lika, och Säkerhetsbolaget ligger lite emellan de två grupperingarna.

I detta kapitel kommer vi följande att presentera våra fynd mer djupgående baserat på analysens kategorier.

4.1 Övergripande strategi och planering

I 2.2.2 beskrivs att riskhantering bör implementeras i enlighet med en verksamhets övergripande strategi. Här visar vi de mönster vi fann gällande hur de undersökta företagens strategi påverkar deras riskhantering samt planering av denna.

Tabell 4.2: Jämförelse av företagens övergripande strategi, som påverkar hur de arbetar med riskhantering.

	Formell riskhantering	**Tilldelning av ansvar**	**Arbetar agilt**	**Typ av kunder**	**Planering av riskhantering**
E-handelskonsultbolaget	✓ (A33, A105)	✓ (A41)	Delvis (A19, A21)	Mid- high-range (A17)	✓ (A27, A30, A38, A60)
Produktutvecklingsbolaget	× (B22, B24)	× (B44, B62)	× (B16)	Från branschledande till startups (B14)	× (B28)
AI-konsultbolaget	× (C38)	× (C51, 68)	✓ (C12, C16)	I utvecklingsfas, bryr sig ej om riskhantering (C20, C59, C68, C70, C76)	× (C24)
Vårdbolaget	✓ (D18)	✓ (D18)	✓ (D14)	Internt (D12)	✓ (D18, D20, D31)
Säkerhetsbolaget	× (E141)	Delvis (E139, E141, E145)	✓ (E49)	20–1000 anställda (E113, E115)	Delvis (E47, E49, E139, E141)

Som tabell 4.2 synliggör är respondenternas företag olika. Endast två av fem har formell och strukturerad riskhantering . Det är också enbart de två största företagen, E-handelskonsultbolaget och Vårdbolaget som har en strukturerad riskhantering. Vidare fann vi att hur företagen arbetade med sina kunder, och vilka typer kunder företagen söker har ett samband med hur företagen arbetar med riskhantering. Produktutvecklingsbolaget och AI-konsultbolaget saknar helt struktur för riskhantering (B22, B24, C38), där Säkerhetsbolaget åtminstone tilldelar visst ansvar för riskhantering (E139, E141, E145). Produktutvecklingsbolaget och AI-

konsultbolaget arbetar båda främst med att skapa “proof of concept”-tjänster (B70, B72, C24, C26) som då inte är färdiga produkter, och därför inte ställer lika höga krav på säkerhet och kvalitet. Även Säkerhetsbolaget beskriver att de just nu har praktikanter som genomför olika projekt som då fungerar lite som ett proof of concept (E38), dock är tanken där att tjänsten som nu provas av kunder är densamma som ska köpas. Både Produktutvecklingsbolaget och AI-konsultbolaget beskriver att deras kunder inte är intresserade av något större riskhanteringsarbete (B82, C30, C42, C59, C76) där AI-konsultbolaget också aktivt söker den typen av kund (C59, C68, C70, C76). Istället prioriterar de att kunna gå snabbt framåt i sina utvecklingsprojekt genom att spendera tid och resurser på rätt saker (B22, B28, B78, C36, C76, C82), vilket även gäller Säkerhetsbolaget (E41, E43, E125, E127). Dessa tre företag upplever också att den kanske största risken är att lägga tid och resurser på fel saker, alltså att investera i en produkt eller funktionalitet ingen vill ha (B42, B60, B78, C32, C76, E41). Vi ser här ett tydligt samband mellan val av kund, typ av tjänst eller produkt som utvecklas och hur den utvecklingen måste gå till. Även företagets storlek verkar spela stor roll, där både Produktutvecklingsbolaget och AI-konsultbolaget säger att om de vore betydligt större skulle de vilja implementera riskhantering, men att det inte är möjligt för dem nu (B42, B60, B88, C57, C59, C68, C70). Detta överlappar i viss mån med Säkerhetsbolagets observation om kunder, där de säger att kunder måste ha minst 20 anställda för att problem med beroendehanteringssystem ska bli tillräckligt stora för att behöva Säkerhetsbolagets tjänst, sedan efter 1000 anställda har kunder generellt tagit fram lösningar internt (E113, E115).

Vi fann också att det inte gick att hitta någon tydlig koppling gällande huruvida företagen arbetade agilt och deras riskhanteringsarbete. Fyra av de fem undersökta företagen arbetar agilt i olika utsträckningar. Vi ser att E-handelskonsultbolaget och Vårdbolaget båda har en struktur som liknar den som föreslås av Nyfjord och Kajko-Mattson (2008), där det finns särskilda avdelningar, roller och en struktur för riskhantering utanför det agila arbetet. De andra tre företagen har som sagt inte formaliserad riskhantering, men det förefaller oss vara relativt smärtfritt för Säkerhetsbolaget att implementera riskhantering genom den integrationsmodell Hammad och Inayat (2018) föreslår för just SCRUM, och då kunna fånga fler risker. Det skulle då inte behöva innebära någon omorganisation utan snarare en tydligare ansvarsfördelning gällande risk, samt göra de aktiviteter som nu sker implicit till explicita i exempelvis sprint planning, och använda register av risker. Vi fann alltså precis som i en undersökning av Hammad et al. (2019) att riskhantering i agila verksamheter ofta kan ske informellt och implicit, men samtidigt ser vi precis som Siddique & Hussein (2014) att agilt arbete inte verkar vara något hinder för att implementera formell riskhantering. Våra fynd indikerar att det snarare rör sig om andra faktorer, där det i grunden handlar om hur de bäst arbetar på ett hållbart sätt för att skapa värde för verksamheten. Men också att på de företag med formell riskhantering sker hanteringen av risker kopplade NPM ofta ändå implicit på individuella utvecklare frånkopplat från övrig riskhantering (A60, D22, D53, D66).

Företagens strategi och grad av riskhanteringsarbete påverkar följaktligen också hur de planerar sin riskhantering. Både E-handelskonsultbolaget och Vårdbolaget fattar i planeringsfasen beslut kring risker och kan planera sitt riskarbete (A27, A30, A38, A60, D18, D20, D31), men de andra tre företagen gör det inte i samma utsträckning. Dock sker det viss sådan planering i Säkerhetsbolagets sprint planning, men det sker informellt (E47, E49, E139, E141).

4.2 Identifiering, analys och prioritering av risker

Här presenterar vi resultat funna i tre kategorier av vår analys - identifiering, analys och prioritering av risk.

Tabell 4.3: Jämförelse av hur företag arbetar med kategorierna identifiering, analys och prioritering. Tabellen inkluderar också vad företagen beskriver som sin främsta risk.

	Aktivitet för identifiering	**Aktivitet för analys**	**Aktivitet för prioritering**	**Främsta risken**
E-handelskonsultbolaget	QA tillsammans med intressenter (A45, A47)	✓ (A27, A60, A35, A47, A64, A79, A97)	✓ (A35, A47, A85)	-
Produktutvecklingsbolaget	× (B22)	× (B22)	× (B22)	Spendera resurser på något som ej fungerar, ej efterfrågat av kund (B22, B28, B72, B80)
AI-konsultbolaget	× (C24, C38)	× (C24, C38)	× (C24, C38)	Investera i något kunder ej vill ha (C32, C76)
Vårdbolaget	✓ DPO ansvarig (D18, D20)	✓ DPO ansvarig (D18, D20)	✓ DPO ansvarig (D18, D20)	Frågor gällande patientdata/ datasäkerhet (D18, D22, D64)
Säkerhetsbolaget	Delvis (E41, E70, E141, E145)	Delvis (E57, E70, E141)	Delvis (E57, E70, E141)	Att inte få kunder, ingen vill använda tjänsten (E125, E127)

Tabell 4.3 visar igen att företagen kan delas upp i tre grupper, där E-handelskonsultbolaget och Vårdbolaget har formella aktiviteter, Säkerhetsbolaget har det i viss utsträckning, där Produktutvecklingsbolaget och AI-konsultbolaget saknar dessa explicita aktiviteter. Först bör vi nämna att respondenten i Vårdbolaget inte är delaktig i dessa aktiviteter (D66), men vi tycker oss kunna utröna från svaren att de existerar utanför hans roll, då det bland annat finns register och dokumentation som pekar på detta (D24, D26, D28, D45). E-handelskonsultbolaget utför dessa aktiviteter främst i QA-avdelningen, men inkluderar även utvecklare och andra intressenter genom exempelvis workshops (A27, A35, A45, A47, A60, A64, A79, A85, A97). Säkerhetsbolaget utför dessa aktiviteter i viss mån då de inte görs explicit eller strukturerat, men samtidigt har en ansvarsfördelning som inkluderar ansvar för risker, och tenderar att utföra dessa aktiviteter om än implicit. Gällande Produktutvecklingsbolaget och AI-konsultbolaget saknas dessa aktiviteter egentligen helt, där litar företagen istället på den individuelle utvecklaren som får ansvara för dessa beslut grundat på förutsättningar eller krav från kund (B24,

B32, B70, B72, C24, C26, C51). Vi ser här igen att det finns ett mönster där val av kund, typ av tjänst eller produkt som företaget säljer påverkar hur de måste arbeta för att skapa värde och därmed påverkar företagets riskhantering. De tre företag som saknar formaliserad riskhantering beskriver att den största risken är att spendera resurser på fel saker, och där kan formella aktiviteter för identifiering, analys och prioritering mycket väl vara fel saker.

4.3 Dokumentation, övervakning och behandling av risk

I detta avsnitt presenteras resultat från de tre kategorierna dokumentation, övervakning, och behandling av risk, från vår analys.

Tabell 4.4: Jämförelse av huruvida företagen använder sig av ett riskarkiv, hur de övervakar risker gällande NPM, samt om och hur de genomför tester och hur de värderar egenutvecklad kod kontra tredjepartskomponenter.

Företag	Riskarkiv	Övervakning	Testning	Egen utveckling vs. tredjeparts-komponenter
E-handelskonsult-bolaget	✓ (A47)	Vulnerability- publiceringar, prestanda (A79, A81, A85, A89)	-	Tredjepartskompo-nenter mer säkert (A99, A101)
Produktutveckl-ingsbolaget	-	-	✓ (B68, B74)	Egen utveckling bättre där det är möjligt (B76)
AI-konsultbolaget	× (C57)	Övervakar system, loggar fel (C53, C55)	Automatiserade tes-ter (C61, C65)	Bättre kod i tredje-partskomponenter, security by obscu-rity vid egenutveck-lat (C86)
Vårdbolaget	✓ (D24, D26, D28)	Dependency- scan-ning (D51, D53, D55, D57)	Har anställda tes-tare (D41)	En avvägning, an-vänder gärna vers-ioner som varit publicerade ett tag (C39)
Säkerhetsbolaget	× (E45)	Använder sin egen tjänst (E72, E74, E80, E115, E117)	Omfattande auto-matisk testning (E90, E92, E94, E96, E98, E100)	Tredjepartskompo-nenter mer säkert (E143, E145)

Vi ser igen i tabell 4.4 att det är de två företagen som har formaliserad riskhantering som då också har en formaliserad funktion i riskarkiv. Det är också de två samt Säkerhetsbolaget som har automatiserat övervakning gällande säkerhetsuppdateringar av sina NPM-paket, där E-handelskonsultbolaget använder liknande tjänster till Säkerhetsbolagets egna. I Vårdbolaget görs denna automatiska kontroll inte i samtliga projekt, men planerar att utföra det i varje projekt. I varje "bygge" görs dock en granskning av rapporterade risker för inkluderade paket

(D51). AI-bolaget har dock inga explicita rutiner för sådan övervakning av NPM-paket, utan det ligger mer under den enskilde utvecklarens ansvar (C49, C51, C53). Den övervakning de har går snarare ut på att se till att system fungerar som de ska. Säkerhetsbolaget litar fullt ut på sin egen tjänst som är framtagen just för denna typ av övervakning, så kallad vulnerability tracking, av tredjepartskomponenter, men är också det företag som kanske har mest rigorös testning av paket, främst automatiserad men till viss del även manuell. Vi bedömer testning som en typ av behandling för att minimera vissa risker vid utveckling med NPM-paket. Det framkom inte under intervjun huruvida E-handelskonsultbolaget utför tester och då hur, men samtliga andra företag beskriver att det görs. Vårdbolaget har anställda vars primära uppgift är att skriva och utföra tester. Dessa tester undersöker främst ifall en version av ett paket är kompatibelt med andra paket, och om funktionaliteten för dessa paket bibehålls vid uppdatering. Det testas alltså inte för säkerhetsrisker, utan snarare att mjukvaran fortfarande fungerar som den ska.

Annars använde företagen flera olika typer av behandlingsstrategier. Exempelvis försöker AI-konsultbolaget avtala bort ansvaret för säkerhet, och därmed minska riskexponeringen (C72, C74). De arbetar också på timbasis för att undvika att få obetalda timmar (C30). E-handelskonsultbolaget kan ofta eskalera beslut till kund, som då får ta besluta hur en risk ska behandlas (A62, A85, A102). Ett tydligt mönster är att fyra av fem undersökta företag vill hålla nere antalet beroenden och paket, och gärna undviker triviala beroenden (A62, B64, B66, C44, C88, C91, D31, D35, D39). Ofta undersöks nyttan av ett paket, och huruvida samma funktionalitet kan göras själv istället (A62, B64, B66, C82, D39). Där tänker Säkerhetsbolaget lite annorlunda, och föredrar i regel att använda tredjepartskomponenter (E143, E145). Detta beror främst på två faktorer, dels så innebär fler beroenden mer arbete med att hantera dessa, dels beror det på hur företagen värderar säkerheten i NPM-paket jämfört med hur säker kod de själva kan skriva. E-handelskonsultbolaget och Säkerhetsbolaget ser tredjepartskomponenter som mer säker kod, där E-handelskonsultbolaget främst väljer att skriva egen kod av prestandaskäl (A99, A101, E143, E145). Vårdbolaget litar också i regel på kodkvaliteten i NPM-paket, men väntar gärna med att uppdatera tills versionen bevisats vara säker (C39). Det företag som sticker ut mest är Produktutvecklingsbolaget som föredrar att skriva egen kod där det är möjligt (B76). Detta är intressant då det är just Produktutvecklingsbolaget som arbetar med kanske minst säkerhetstänk gällande NPM-paket. AI-konsultbolaget resonerar om att båda infallsvinklar kan ha sina fördelar, då egenutvecklad och då unik kod kan skydda mot automatiserade attacker som annars är en risk vid användning av NPM-paket, även om den egna koden är av sämre kvalitet (C86).

4.4 Best practices för utveckling med open source & NPM

Tabell 4.5: Best practices och åtgärder, samt efterlevnad.

	E-handelskonsultbolaget	Produktutvecklingsbolaget	AI-konsultbolaget	Vårdbolaget	Säkerhetsbolaget
Använd kända, säkra komponenter	Delvis (A33)	-	-	-	-

Övervaka källan/communityn	-	-	-	✓ (D80)	Snart (E70)
Övervaka antagna komponenter	✓ (A79, A81)	-	× (C49)	✓ (D51)	✓ (E45)
Installera uppdateringar skyndsamt	Endast säkerhetsuppdateringar (A66, A79)	× (B46)	✓ (C63)	I vissa projekt (D53)	× (E76)
Dedikera personal med ansvar för analys och övervakning av komponenter	✓ (A89)	× (B62)	× (C49)	× (D51)	× (E49)
Använd privata paketregister	Delvis, endast för vissa kunder (A70)	× (B42)	-	Privat paketregister endast för egenutvecklade paket (D47, D49)	× (E61, E65)
Granskning av källkod	Ibland, men inte av säkerhetsskäl	Ibland, men inte av säkerhetsskäl (B36)	-	×	×

Företagens användning av privata paketregister tycks korrelera med deras storlek. Endast de två största, E-handelskonsultbolaget samt Vårdbolaget (A70, D61, D65), uppger att de använder privata paketregister i dagsläget. Produktutvecklingsbolaget menar att det är en fråga om antal anställda och företagets ekonomiska resurser, samt att kostnaden i deras fall inte skulle motiveras av de eventuella vinsterna (B42). Endast e-handelskonsultbolaget använder sitt privata paketregister för att lagra kopior av offentliga paket, de uppger dock att detta främst är p.g.a. att vissa kunder har byggservrar som inte får vara kopplade till internet, det handlar alltså inte i första hand om att säkerställa fortsatt tillgång till paket i händelse av avpublicering (A72).

Samtliga tillfrågade företag tillämpar någon form av granskning, dvs riskanalys, av nya paket. Denna granskning tycks dock vara informell i stor utsträckning, och sköts i första hand av utvecklare snarare än att ingå i verksamhetens övergripande riskhanteringsarbete, i de fall då ett sådant finns.

E-handelskonsultbolaget uppger sig t.ex. bedriva ett ganska omfattande riskhanteringsarbete, där varje projekt inleds med en förstudie och workshops med berörda intressenter (A47), där bl.a. risker diskuteras. De har även en QA-avdelning, projektledare, leverans- och kvalitetsansvariga som ofta hanterar säkerhetsaspekter och arbetet kring dem (A41, A105). Om en risk identifieras och bedöms som allvarlig eskaleras den till kunden, som får ta ställning till huruvida den är acceptabel eller ej (A103). Trots detta känner respondenten inte till några exempel där NPM-paket diskuterats i dessa sammanhang (A33). Man har visserligen ett antal "etablerade" paket som ofta används (A33, A77), men beslut om, och granskning av, nya NPM-paket sker i regel löpande under projektets gång, ofta i samband med implementation av specifika funktioner där paketet används (A27, A60). Man använder sig också av s.k. code

reviews, vilket innebär att kod granskas av flera utvecklare innan den driftsätts, enligt respondenten är analys av paket ett delmoment i denna granskning (A62).

Vårdbolagets företrädare beskriver ett liknande arbetssätt, som i första hand präglas av bestämmelserna i GDPR (D18). Företaget har en personuppgiftsansvarig som hanterar risk- och säkerhetsfrågor. Den personuppgiftsansvariga kommunicerar med utvecklare och arkitekter, och påverkar dem i viss utsträckning (D18), dock inte gällande val av teknik (D22). Man lägger stor vikt vid dokumentation av risker och fattade beslut, då GDPR kräver detta (D28) och uppger sig inte ta några medvetna risker överhuvudtaget när det kommer till tredjepartsverktyg (D83). Likt e-handelskonsultbolaget tycks Vårdbolaget sakna en formaliserad process för granskning utav paket, detta sköts istället av utvecklare som ett delmoment i de rutiner som finns för granskning av kod.

De övriga tre uppger sig inte ha något formaliserat riskhanteringsarbete (E49, E121, C38, B22). Produktutvecklingsbolaget tycker sig inte ha råd att prioritera risker kopplade till NPM-paket överhuvudtaget (B22, B60, B78). Trots detta verkar dessa företag i praktiken tillämpa en liknande granskning som E-handelskonsultbolaget och Vårdbolaget, där riskbedömning av NPM-paket utförs av utvecklare under projektets gång. Säkerhetsbolaget använder, likt Vårdbolaget och E-handelskonsultbolaget, sig utav en formaliserad process för code reviews, nya paket granskas därmed av mer än en utvecklare. Produktutvecklingsbolaget säger sig ha en typ av informell code review, där utvecklare granskar nya paket i samband med att de installerar dem (B62).

När det kommer till vilka kriterier som används vid granskning av NPM-paket tycks det finnas stora skillnader mellan de undersökta företagen. Samtliga uppger att de lägger stor vikt vid paketets användare, ett paket som används av många anses vara pålitligt, särskilt om stora, etablerade företag och organisationer finns bland dessa användare. Företagen tycks även lägga stor vikt vid paketets avsändare, med undantag för Säkerhetsbolaget, som istället framhäver vikten av god dokumentation och regelbunden aktivitet.

Tabell 4.6: Kriterier för granskning av paket.

	E-handelskonsultbolaget	**Produktutvecklingsbolaget**	**AI-konsultbolaget**	**Vårdbolaget**	**Säkerhetsbolaget**
Innehåll/ källkod	Delvis, ej av säkerhetsskäl. (A68)	Delvis, granskar run-scripts (B28)	-	× (D37)	× (E53)
Ålder / väl etablerat	-	✓ (B34)	-	✓ (D31)	-
Andra användare (antal, etablerade eller ej)	✓ (A62)	✓ (B64)	✓ (C44)	✓ (D35)	✓ (E51)
Dokumentation	-	-	-	-	✓ (E55)

Leverantör / Underhållare / sponsorer (“avsändare”)	✓ (A62)	✓ (B34)	✓ (C44)	✓ (D35)	× (E51)
Aktivitet (regelbundna uppdateringar, inte övergivet)	-	-	-	✓ (D35)	✓ (E47)
Komplexitet / Överflödig kod	✓ (A68)	✓ (B66)	-	✓ (D39)	-

Dessa resultat i tabell 4.6 tyder på en avsaknad av samsyn i branschen gällande vilka kriterier ett paket bör utvärderas utifrån. Det är uppseendeväckande att Säkerhetsbolaget, som kan anses vara något utav en auktoritet på området, inte lägger någon större vikt vid paketets avsändare, något de andra framhäver som viktigt. Samtidigt tycks de övriga, med undantag för Vårdbolaget, inte anse att aktivitet är ett viktigt mått, medan Säkerhetsbolaget anser att det är helt avgörande. Det kan finnas ett samband mellan ett pakets ålder och mängden historisk aktivitet, vilket kan innebära en falsk trygghet då historisk aktivitet inte är någon garanti för ett “levande” paket. Säkerhetsbolaget adresserar detta genom att även inkludera den långsiktiga trenden i sin analys (E47). E-handelskonsultbolaget, Produktutvecklingsbolaget och Vårdbolaget säger sig undvika paket som innehåller kod som är onödigt komplex eller överflödig för det tänkta användningsområdet (A68, B66, D39).

Vårdbolaget, Säkerhetsbolaget samt E-handelskonsultbolaget uppger att de integrerat olika typer av automatiserad så kallad “dependency scanning” i sina CI (continuous integration)-pipelines. Detta innebär bl.a. att de olika utvecklarnas bidrag till kodbasen genomgår automatiserade tester i samband med att de laddas upp i det gemensamma versionshanteringssystemet. Dependency scanning-verktyget liknar den inbyggda NPM-funktionen “audit” och kontrollerar att inga beroenden i projektet har kända sårbarheter, om sårbarheter upptäcks misslyckas integrationen varpå utvecklaren får åtgärda bristerna och försöka igen. Sårbarheter ska publiceras på ett ansvarsfullt sätt, vilket innebär att det i regel finns en uppdatering tillgänglig vid tidpunkten för offentliggörandet (Npm inc, 2020d). De övriga företagen, AI-konsultbolaget och Produktutvecklingsbolaget, anser sig inte ansvara för drift eller löpande underhåll av några utsatta system i dagsläget, de menar att dessa frågor i många fall hanteras av kunden i ett senare skede.

Även strategin gällande ej säkerhetsrelaterade uppdateringar skiljer sig mellan de olika företagen. De flesta av dem uppger att de försöker hålla sina paket uppdaterade såg gott det går, särskilt när det gäller stora paket med hög grad av sammankoppling till andra paket i projektet. Vårdbolaget är det enda av de undersökta företagen som aktivt väntar med att uppdatera, de anser att det är bättre att vänta tills den nya versionen beprövats och visat sig vara stabil och säker (D68, D80). Säkerhetsföretaget instämmer i att detta tillvägagångssätt kan ge ett visst skydd, även om de själva inte tillämpar det (E78).

Samtliga tycks vara överens om att en underlåtenhet att uppdatera under längre tid riskerar att skapa s.k. teknisk skuld. Detta beror på att många uppdateringar inte är helt bakåtkompatibla, och därför kan kräva ändringar i den befintliga kodbasen. Även om dessa ändringar ofta är små, kan den sammanlagda effekten av många uppdateringar bli stor, vilket leder till att stora

förändringar krävs när man väl väljer att uppdatera efter en längre tids inaktivitet. Både Produktutvecklingsbolaget och E-handelskonsultbolaget gör undantag för små paket, som utför en isolerad funktion utan beroenden.

Även om NPM medför en problematik är det tydligt att samtliga respondenter ser det som en nödvändighet, där det inte finns några bra alternativ (A99, B74, C82, C84, D100, D102, D104, E109). Utveckling för webb kräver idag nästan användning av Javascript, som i sin tur kräver användning av NPM.

5 Diskussion

I diskussionen lyfter vi kortfattat de viktigaste fynden och sätter dem i en bredare kontext, där vi problematiserar och diskuterar deras innebörd.

5.1 NPM – en nödvändighet trots brister

Trots den problematik som beskrivs i studien tycks respondenterna och branschen i stort vara överens om att NPM och liknande ekosystem är här för att stanna. Vinsterna i form av tidsbesparingar och bättre slutprodukter överväger nackdelarna med råge. Moderna webbplatser och applikationer är oerhört komplexa och det är inte realistiskt att börja från grunden i varje enskilt projekt eller organisation. Det förefaller inte heller vara möjligt för en enskild leverantör att skapa en heltäckande helhetslösning för dessa behov i form av ett ramverk eller dylikt. Utvecklingen mot applikationer bestående av mer granulära komponenter från ett större antal leverantörer kommer sannolikt att fortsätta, till och med företag som t.ex. Microsoft som traditionellt sett betraktats som allt annat än öppna satsar nu stort på samarbete och öppen källkod, och äger nu både GitHub och Npm inc (Friedman, 2020; Microsoft, 2018). Det är viktigt att ha i åtanke att NPM och liknande ekosystem fortfarande är ett relativt nytt fenomen, många av problemen som avhandlas kan betraktas som barnsjukdomar, och kan sannolikt lösas med en kombination av teknik och kunskap.

5.2 Övergripande riskhanteringsarbete och bakomliggande faktorer

Det var bara två av fem undersökta företag som hade implementerat ett formaliserat riskhanteringsarbete i sin verksamhet. Vi såg ett tydligt mönster mellan företagets storlek och förutsättningar, samt strategi för hur företaget ska konkurrera, och med hur de hanterar risker. Större verksamheter har en annan kapacitet både ekonomiskt samt gällande personal och marginaler för att implementera riskhantering. Det kan kräva nya roller, processer eller till och med avdelningar som helt enkelt inte är lika aktuellt för mindre företag. Två av de undersökta företagen hade ingen formaliserad riskhantering alls, dels på grund av sin storlek, men också för att den typ av tjänst de utvecklade inte kräver det i lika stor utsträckning. Samtidigt var deras kunder inte heller intresserade av att varken betala för eller vänta längre för produkter som utvecklats med god riskhantering. Det är därför svårt att klandra dem för att de inte spenderar resurser på riskhantering. De två företag som har implementerat riskhantering var de två största, som har kapaciteten och resurserna för att göra detta på ett bra sätt. Det är också de två företag vars produkter och kunder (notera att Vårdbolaget inte har en kund, utan utvecklar system och tjänster för internt bruk) i större utsträckning kräver god riskhantering.

Det ska också nämnas att i de företag som saknar formaliserad riskhantering är det inte fritt fram att göra vad som helst, och de är inte heller helt omedvetna om risker, vare sig övergripande eller specifika för NPM. Men vi fann att explicita strukturer och formella kärnaktiviteter där saknats. De företagen identifierar och utvärderar fortfarande risker. Det görs dock i mindre utsträckning där ansvaret faller på individen och kommunikation om risker uteblir, vilket leder till sämre kontroll och hantering än om de identifierade kärnaktiviteterna och best practices implementerats.

5.3 Riskhantering enligt modell, men ej för NPM

Vidare fann vi att även i de företag som implementerat de kärnaktiviteterna gällande riskhantering inte arbetade lika formaliserat gällande just NPM. Även om strukturen finns tas ofta beslut gällande beroenden på en låg nivå individuellt och ofta enligt informella och implicita riktlinjer. Vissa beslut som vilka paket som ska användas tas i några av företagens planeringsfas, och tyder då på att identifiering och analys av risk gällande NPM-paket utförs i någon form. Men överlag sker beslut och bedömning av NPM-paket utanför det formaliserade riskhanteringsarbetet. Vi ser det som ett problem att detta inte tas på ännu större allvar i samtliga undersökta företag. Det är rimligt att utvecklare är de i organisationen som har störst kunskap för att fatta dessa beslut, men vi saknar en tydligare ansvarsfördelning där det finns en ansvarig för granskning och bedömning som då också ansvarar för att ta fram riktlinjer och rutiner för beroendehantering. Vi menar att det finns möjligheter för samtliga verksamheter att med små grepp införa en sund riskhantering, exempelvis genom tilldelning av ansvar, men också genom att låta problem kring riskhantering av NPM-paket få en explicit plats på dagordningen för att införa någon form av strukturerad identifiering, analys, övervakning och behandling.

5.4 Brist på samsyn

Vi fann en utbredd medvetenhet om problemen kopplade till NPM, alla tillfrågade organisationer förefaller förstå att det är ett potentiellt stort problem, trots detta verkar det saknas en samsyn kring många aspekter. Det finns givetvis ingen universallösning som passar alla verksamheter, men det är uppseendeväckande att t.ex. kriterierna som används för bedömning av paket skiljer sig så markant. Oavsett vilken typ av verksamhet det rör sig om så bör det gå att enas kring en gemensam modell för denna riskanalys. Ytterligare forskning behövs för att fastställa vilka metoder som kan vara lämpliga för olika verksamheter och branscher.

Vi fann också vissa missuppfattningar gällande hur man riskerar att drabbas. Några av de tillfrågade företagen menar t.ex. att det är mer eller mindre riskfritt att använda tredjepartskomponenter i vissa sammanhang, t.ex. i funktioner som inte exponeras mot internet, eller i klientapplikationer som körs i en webbläsare. Denna bedömning tycks inte vara i linje med forskning på området, skadlig kod som skrivits med ont uppsåt behöver inte anropas utifrån för att orsaka stor skada, och även om webbläsare ofta har säkerhetsfunktioner som ger ett visst skydd kvarstår stora risker, t.ex. informationsläckor som en följd av xss (cross site scripting)-attacker.

Ett av företagen uppgav att de inte tar några risker alls när det kommer till tredjepartskomponenter, trots att de beskriver problemet med skadlig kod i paketregistret som utbrett. Det går givetvis att minimera riskerna i stor utsträckning, men i dagsläget finns det anledning att ifrågasätta om det verkligen går att freda sig helt och hållet. Ett sådant synsätt riskerar att leda till passivitet. IT-säkerhet är en evig kapplöpning, och det kommer alltid finnas risker som ännu ej upptäckts, vilket gör det nästintill omöjligt att lämna absoluta garantier.

5.5 Övervakning av sårbarheter, säkerhetsuppdateringar och automatiserade verktyg

De flesta tillfrågade företagen verkar förlita sig på tekniska lösningar, främst så kallad vulnerability tracking och dependency scanning, för att hålla sig ajour med nya säkerhetsråd och uppdateringar. Dessa system kan användas för att säkerställa att tillgängliga säkerhetsuppdateringar installeras och ger därför ett ganska gott skydd mot kända problem. Det är dock viktigt att understryka att de i dagsläget inte ger något skydd mot sårbarheter som ännu inte upptäcks eller rapporterats. Forskning och historiska erfarenheter visar att så inte alltid sker direkt, ett exempel på detta är den tidigare nämnda event-stream-incidenten, där skadlig kod förblev oupptäckt i 46 dagar. Vi ser en viss risk att dessa system kan invagga användaren i en falsk trygghet, som skulle kunna leda till att man är mindre noggrann i sitt övriga arbete med NPM-paket. Mer sofistikerade lösningar har föreslagits, t.ex. mjukvara för att automatiskt granska källkod. Även enklare förändringar i beroendehanteringssystemet, som exempelvis tydligare information om beroendedjup, underhållare och aktivitetstrender hade kunnat underlätta granskning av paket. Problemet är i grunden en fråga om tillit och ansvarsfördelning, vilket gör att tekniska verktyg sannolikt bara är en delmängd av en potentiell lösning.

5.6 Studiens tillförlitlighet

Validiteten i undersökningen kan ifrågasättas. Vi valde en kvalitativ undersökning med semistrukturerad intervju för att kunna försäkra oss om att frågor och svar uppfattats på rätt sätt. Detta innebar dock ibland följdfrågor som kan ha varit något ledande. Vi upplevde det vara en svår balans att försäkra att respondenten uppfattat frågan korrekt utan att göra det för tydligt vilket typ av svar som önskades. Ett annat möjligt problem är huruvida respondenterna verkligen kan tala för hela företaget. Det är möjligt att ett företag har fler processer, rutiner eller strukturer än vad respondenten är medveten om. I vissa fall verkar respondenterna ha varit i god position för att ha bra inblick i hela verksamheten, men i andra fall har de varit i en sämre position för att säkert kunna avgöra exakt hur företaget arbetar inom samtliga områden. I vår analys har vi därför hållit oss ifrån att göra större påståenden än vad empirin gett oss belägg för.

En motivation bakom beslutet att använda semistrukturerade intervjuer var att de är lättare att replikera än ostrukturerade intervjuer, och då också kan bidra till högre reliabilitet. En semistrukturerad intervju är dock inte det bästa alternativet för undersökningsmetod om målet enbart gäller hög reliabilitet, utan har där större brister än exempelvis en enkätundersökning. En annan faktor som påverkar studiens reliabilitet och tillförlitlighet är undersökningens urval, där respondenter från fem företag inte kan påstås vara statistiskt signifikanta. Därför kan slutsatser från denna undersökning inte menas gälla för samtliga företag som använder NPM vid mjukvaruutveckling. Vi presenterar dessa fynd ödmjukt och understryker behovet av vidare forskning.

Urvalet bestod också av respondenter från företag i olika storlekar som arbetar i olika delar av industrin. Detta var ett medvetet val för att undersöka om just dessa faktorer kunde påverka verksamheters riskhantering både i stort och gällande NPM. Men det innebär samtidigt en sämre reliabilitet just då företagen är så olika, där en undersökning av företag med mer lika förutsättningar hade inneburit ett mer tillförlitligt resultat.

Vi är också medvetna om att Säkerhetsbolaget har en annorlunda ställning jämfört med de andra undersökta företagen, och kanske inte kan sägas vara representativa för företag verksamma inom mjukvaruutveckling generellt. Vår tanke var från början att tidigt i studien genomföra en intervju med Säkerhetsbolaget som en del av en grundad och induktiv undersökning, och därefter använda eventuella fynd som ett verktyg för vårt teoriskapande. På grund av både etiska skäl där Säkerhetsbolaget kan tänkas ha ett intresse av att understryka säkerhetsproblem vid utveckling av NPM och styrkan av deras tjänst, samt svårigheter att schemalägga intervjun behandlades de istället på samma sätt som de andra respondenterna.

Då undersökningen genomfördes under Covid-19-pandemin hade vi inget annat val än att utföra intervjuerna på distans. I ett fall tvingades vi utföra intervjun helt utan video, och i ett annat fall var en av två respondenter utan videolänk. I dessa intervjuer tycker vi oss kunna se att tydligheten i kommunikationen begränsas, och de har betydligt fler tillfällen där talare avbryter varandra. Det vore betydligt bättre om de intervjuerna hade kunnat genomföras med video, och ännu bättre om de utförts på plats för att få högre tillförlitlighet.

5.7 Källkritik

För primärkällor till teoribildning har enbart källor som genomgått peer review använts. Dessutom har vi tittat på publikationens tillförlitlighet, källor artikeln i sin tur använt, och mängden källhänvisningar till artikeln. För metod har vi använt två källor, främst Oates (2006) som specifikt behandlar forskning gällande informationssystem men även tagit stöd från Bryman (2012) som dock behandlar “Social research methods”. Även om denna gäller metoder i ett annat forskningsfält upplevde vi att Bryman (2012) ofta erbjöd en djupare detalj och problematisering av metoder. För källorna i inledningen som inte är teoriskapande, utan bara används för att beskriva problemområdet och bakgrunden fanns sällan lika kvalificerade källor att tillgå. Där använde vi förstahandskällor, till exempel dokumentation och tjänster framtagna för att ta fram data gällande NPM. Vi menar att dessa källor ändå är tillförlitliga för det syfte där de används, som främst är att befästa inom domänen allmänt accepterade fenomen eller siffror. Dessa typer av källor används återkommande i vetenskapliga artiklar på området.

En begränsning var att vi inte kunde få tillgång till standarden ISO 31000:2018 för riskhantering eller IEC 31010:2019 för riskutvärdering. Istället fick vi nöja oss med ett äldre mindre detaljerat gratisalternativ i ISO/IEC 16085:2004, som har reviderats sedan dess publikation. Vi anser att den ändå är relevant i vår användning av den som källa och att de aktiviteter som identifieras där stöttas av andra modeller och källor. Även om ISO/IEC 16085:2004 är en standard är den framtagen i samarbete med IEEE och genomgått peer review.

6 Slutsats

6.1 Forskningsfråga 1

Hur arbetar systemutvecklingsföretag för att hantera de risker som kan uppkomma vid utveckling med beroendehanteringssystemet NPM?

Vi fann att företagen inte använde någon etablerad riskhanteringsteori i arbetet med just NPM, även där den finns övergripande. Arbetet med NPM är snarare frånkopplat från eventuell riskhantering i resten av företaget och sker oftast implicit istället för formaliserat. Det finns en medvetenhet om problematiken, men samsyn om den samt i vissa fall kunskap saknas. Generellt har företagen ändå identifierat många risker, utvärderar individuella paket och har någon form av övervakning av inkluderade paket. Men sammantaget finner vi många brister där ansvarsfördelning och riktlinjer saknas, och risken för attack via NPM-paket underskattas. De undersökta företagen riskerar att ha dålig kontroll av sitt riskhanteringsarbete och missa mycket gällande NPM.

6.2 Forskningsfråga 2

Vilka faktorer föranleder vald strategi för riskhanteringen?

Vi fann att kapaciteten för att implementera formaliserad riskhantering och kostnaden av en sådan implementation var en grundläggande faktor. Andra faktorer är val av kund och vad den kunden efterfrågar och prioriterar, samt vilken typ av tjänst eller produkt som utvecklas. I vissa fall krävde omständigheterna god riskhantering, och där fanns även kapaciteten och resurserna för att ha ett formaliserat arbete. I andra fall fanns inte dessa krav, och inte heller möjligheten för företag att implementera formell riskhantering på ett kostnads- eller tidseffektivt sätt. Det är för oss tydligt att företagens riskhanteringsstrategi går i linje med deras övergripande strategi, även om vi menar att de underskattar och undervärderar vissa av de risker NPM medför.

6.3 Förslag till vidare forskning

Vi ser ett stort behov för mer forskning på området. Dels krävs en mer omfattande studie med fler respondenter för att kunna avgöra signifikansen av våra fynd samt öka medvetenheten om problemområdet i både industrin och akademin. Vi tycker också att det behövs förslag för hur paket utvärderas, och modeller och ramverk för hur ett paket och dess beroenden ska kunna bedömas på ett tillförlitligt vis. Vi ser det som ett problem att säkerhetsråd och säkerhetsuppdateringar enbart kan reparera existerande sårbarheter, där behövs också bättre metoder för hur verksamheter kan arbeta proaktivt snarare än reaktivt med tanke på att det kan ta flera år innan säkerhetshål täpps till. Vi avgränsade oss från att beröra problem gällande licenser samt vem i kedjan som bär ansvar i någon större utsträckning, men fann att där behövs forskning för att undersöka hur företag hanterar detta problem och strukturerar kontrakt.

Appendix A

Transskript E-handelskonsultbolaget
30/4–2020, via Microsoft Teams
Längd: 43 minuter

Deltagare och förkortningar:
Författare: David Strömbäck (**DS**), Hannes Carlsson (**HC**)
Respondent: Full-stack-utvecklare på E-handelskonsultbolaget (**FS**)

Rad	Talare	Tal
1	HC	Vi tänkte att innan vi börjar med... vi börjar lite med övergripande med grundläggande frågor om dig och företaget. Så det första vi ska avklara är om du önskar anonymiseras med en pseudonym i uppsatsen, t.ex. respondent 1,2,3,4, och då också om [***] ska anonymiseras som företag A eller liknande?
2	FS	Bra fråga, jag tror att både jag och mitt företag bör anonymiseras, mest för att jag inte riktigt kan svara på frågan.
3	HC	Då gör vi så, det blir bra. Och sen... Om du helt enkelt skulle kunna beskriva er verksamhet lite övergripande?
4	FS	Yes. (Säg till om ni hör gräsklipparen för mycket för då stänger jag fönstret). Jo, [***], vår verksamhet sysslar med att bygga e-handelssystem och ge advice och implementation och stöd till våra kunder som primärt sysslar med e-handel. Vi säger att vi är "den digitala partnern for modern commerce", så fort du har någonting med digital Commerce att göra så kan vi hjälpa dig på flera fronter, både rådgivning, olika system och då själva hemsidan också. Så det är kort vad vi gör.
5	DS	Alright
6	HC	Ja, så att då kan man säga att systemutveckling är en kärnverksamhet?
7	FS	Absolut, kollar man ur bolagets perspektiv så är majoriteten av vår personal utvecklare och vi bedriver vår verksamhet som en konsultverksamhet så att våra partners, våra kunder då, säljer vi ofta projekt, fastpris eller i timmar. Så att man... man kan se oss som IT-konsulter också men med en nisch på e-handel.
8	HC	Ja, bra, sen då också bara lite information om dig och företaget. Vilken roll har du på företaget?
9	FS	Mm, jag är utvecklare, jag har haft lite olika roller, och sen.… vi har inte så mycket... vi har ganska platt hierarki så jag sitter som arkitekt i vissa bitar, jag sitter som rådgivande part i vissa projekt, men min faktiska roll är "Commerce system developer".
10	HC	Ja, okej
11	DS	Ja
12	HC	Och hur stort är företaget?

13	FS	Personalmässigt så tror jag vi är någonstans [***] anställda, och ja... vad omsätter vi? [***] miljoner kanske. Vi har kontor i [***] som är utvecklar-kontor, där det finns mer anställda. Sen har vi lite mindre säljkontor och annat i [***], och ett litet i [***].
14	HC	Just det.
15	FS	Precis, men det är inget utvecklingskontor för närvarande utan sälj.
16	HC	Och vilken typ av kunder har ni ungefär?
17	FS	De säljer ju saker, vi brukar satsa på "mid" till "high range" av kunder. Så att exempelkunder är [***], det segmentet av kunder. De absolut största brukar lösa det här helt internet för dem är så pass stora att de kan ha hela avdelningar för detta, exempelvis [***]. Men även med dem största kunderna kan det hända att vi hjälper till med något projekt, rådgivning eller liknande.
18	HC	Ja, och sen, också en kontrollfråga, skulle du säga att ni arbetar agilt?
19	FS	I den mån vi kan, ja. Skulle jag läsa igenom det agila manifestet och jämföra mitt vardagliga arbete: Nej. Men vi har det i tankarna.
20	HC	Ja, precis. Så i någon utsträckning?
21	FS	Absolut. Vår tanke är ju att göra det, vår vilja är att göra det. Men verkligheten, kundkrav, projektstruktur ibland tvingar fram saker som agil-bibelläsare inte hade kallat för agilt.
22	HC	Nej, precis. Och då också, sista kontrollfrågan här, använder ni NPM-paket i er utveckling?
23	FS	Det gör vi, både våra egna och saker som är publicerade på offentliga registryn.
24	HC	Tusen tack, då ska vi gå vidare här. Då tänker vi att vi gör så att vi beskriver lite scenarion, och så kan du beskriva, i ett sådant scenario, ungefär vad ni kommer göra och hur det funkar hos er.
25	FS	Yes
26	HC	Ett scenario då. Det kan ju börja med att ni ska dra igång ett nytt utvecklingsprojekt. Och då undrar vi lite, hur ser riskhanteringsarbetet ut i ett initialt skede? Alltså vilka aktiviteter finns, planering, identifiering, analys, prioritering, övervakning i själv riskarbetet? Vet du det?
27	FS	Vi har... om jag tar ett exempelprojekt helt hypotetiskt. I förstudien adresserar man en hel del av de här bitarna, man analyserar vilka externa komponenter man använder, vilket externa bibliotek man skulle använda och adresserar det tillsammans med kunden, och liksom vilka risker det medför. På workshops tillsammans med kunden. Sen löpande under projektets gång kan det vara så att krav kommer upp, det är kanske change requests eller andra krav som (vi) säger att amen... det här kan vara rätt enkelt att lösa med ett NPM-paket, och då sköts riskhanteringen i det ärendet i sig, isolerat. Hur man ska ta hänsyn till... ja, vad man ska göra och vad man ska tänka på.
28	HC	Ja precis
29	DS	Det här riskhanteringsarbetet då specifikt gällande NPM-paket, har det någon koppling till företagets övergripande strategi när det gäller riskhantering, och är det andra moment involverade i det arbetet förutom just det här med beroenden och tredjepartskomponenter?

30	FS	Det är det, vi har inte specifikt kring NPM -paket, jag skulle inte säga att det är den största vikten vid vår riskhantering överhuvudtaget, utan det är generell riskhantering för hela projektet, för kunden, ekonomiskt liksom allt sånt. Men då tas även tredjepartsleverantörer in, och där, som en delmängd av dem, så finns liksom NPM -paket. Så att, jag skulle säga att det är en väldigt liten del av vår riskhantering idag.
31	DS	Alright
32	HC	Jo precis, så att då har ni ändå en strategi för riskhantering? Alltså att ni identifierar risker, ni kommunicerar om dem, ni kanske tom analyserar och prioriterar dem?
33	FS	Precis, exakt. Men så vitt jag vet, i dem projekten jag varit med i, så har det aldrig landat i att ett NPM-paket skulle komma upp till diskussion eller något sånt utan vi har ofta rätt etablerade vilka vi använder och sen, de flesta casen, de kommer upp i user stories i projektet. Att okej, nu vill vi lösa detta kravet, och med det kan vi dra in ett NPM -paket, och då tar vi diskussionen då.
34	HC	Förresten då, jag är också liten nyfiken på, förutom NPM då, en vanlig risk, säg risken att gå över budget eller blåsa en deadline, vad det nu kan vara för risker. Då prioriteras dessa, vet du hur det går till? Är det via någon form av matris, algoritm, värden som kommer ut, hur ser det ut?
35	FS	De jag varit med i, och jag har för mig att vi (alltid) gör en liknande approach, är att vi... vi skriver upp alla risker vi kan komma på, grupperar dem i olika logiska grupperingar, och sen värderar dem utifrån då hur "likely" det är att den kommer utfalla, hur stor effekt den har om den utfaller och så får man då ett "score" per risk. Och utifrån... så går vi igenom alla scores, och de som är absolut högst ser vi om vi kan adressera på något sätt, eller om det är en risk vi helt enkelt lever med, men liksom att ha det i bakhuvudet.
36	HC	Ja exakt, tusen tack.
37	DS	Skulle du eventuellt kunna ge något exempel på hur den övervägningen där, alltså risken att något inträffar kontra konsekvensen. Hur den övervägningen kan se ut för ett NPM-paket? Jag antar att ni bedömer risken som ganska låg?
38	FS	Precis, jag har inte varit med i någon riskanalysprocess där ett NPM -paket varit med som en enskild risk, utan det har då varit tredjepartsinteraktioner, men oftast så brukar de vara... det kanske kan vara en större komponent som inte levererat ännu och då ser vi en risk för leveransen, men om det är en komponent som existerar så brukar det vara väldigt liten risk för... ja... det brukar sällan komma upp till diskussion att vi ska integrera det.
39	DS	Ja, yes
40	HC	Sen undrar jag vidare också, tilldelas ansvar inom riskhantering? Har ni roller inom riskhanteringsarbetet? Är någon ansvarig för att se till att kommuniceras, att vissa saker följs, för övervakning eller något sådant?

41	FS	Det beror lite på hur man ser på det, per projekt så finns det en leveransansvarig, som är en roll som är ansvarig för leveransen av hela projektet, och det innefattar att kommunicera till kunden om de olika punkterna som vi gör under projektet, delvis då riskhantering och liksom vilka saker vi ser som risk. Det delas rätt mycket med projektledaren också, under arbetet när man tar fram de olika riskerna. Så det är liksom fronten till kunden, att vi har en leveransansvarig och en projektledare som hanterar det mot kunden. Internt skulle jag inte säga att vi har någon officiell roll för att hantera dem olika riskerna utan det ansvaret ligger på projektledaren att liksom ta med dem i arbetet och adressera dem om det skulle behövas.
42	HC	Kan det finnas fall då man identifierar en risk, vad det nu skulle kunna vara, alltså utanför NPM-paket. Och att den risken bedöms vara så pass stor att man måste strukturera om projektet eller göra saker ganska annorlunda?
43	FS	Oftast inte skulle jag säga, jag skulle säga att en sån risk... upptäcker man en sån risk så har man missat en hel del i förarbetet i liksom... vad är det man ska åstadkomma? Sådana bitar, det bör inte komma som en chock. Det kan påverka hur man bygger någonting, hur man kodar någonting, och det kan till och med påverka "vilket NPM-paket ska vi använda?" eller så. Men, jag kan inte tänka mig att det skulle förändra strukturen på projektet i så fall har vi gjort något dumt innan dess liksom.
44	HC	[skratt] Ja, och jag undrar också, det låter som att ni använder brainstorming som en del i hur ni då identifierar risker?
45	FS	Precis
46	HC	Och då antar jag att ni har... dels av egen erfarenhet, ni har gjort sådana här projekt tidigare, att ni då kommer ihåg, vilka var riskerna då? Men har ni något officiellt arkiv eller lista på risker som ni brukar titta på och ta in, uppdatera osv?
47	FS	Dessa risk-workshopar faciliteras av en från quality assurance teamet som vi har på [***], och jag tror dem har lite sådana matriser och mallar för... Det här är de vanligare, så går vi igenom dem, så är det fortfarande så att vi kan brainstorma. Vi har utvecklare, vi har projektledare, vi har liksom hela gänget med på en sån workshop och så kan dem identifiera andra risker.
48	HC	Ja, okej
49	FS	Nu är jag inte säker, men i dem jag varit med i har det varit rätt förarbete, så jag misstänker att de har någonting.
50	HC	Ja, precis. Sen har vi också en fråga här, men den har vi väl fått lite svar på, huruvida det fattas beslut kring arkitektur och val av tredjepartskomponenter i detta skede? Men det kan det eventuellt göra då?
51	FS	Ska hämta en leverans bara, återkommer om någon minut, sorry.
52	HC	Ingen fara
53		[Paus]
54	FS	Alright, I'm back!
55	DS	Om du vill äta ifred så hade vi ju kunnat pausa ett tag?
56	FS	Nej, det är ingen fara
57	DS	Alright
58	FS	Så... förra frågan... Hannes, du sa någonting där som jag har glömt?

59	HC	Ja precis, det jag undrar är helt enkelt. I det här skedet då, där det liksom är i stort sett planeringsfas fortfarande. Fattas då beslut kring arkitektur och tredjepartskomponenter?
60	FS	Ja, dem stora bitarna, återigen det här för NPM skulle jag inte säga att... De tas inte ens upp för de löses i så fall när man inser ett behov av att det här kan vi ta in i ett senare skede.
61	DS	Precis, så följdfrågan där, om en enskild utvecklare kommer på att det här paketet hade ju kunnat spara en massa jobb. Vilken process behöver de gå igenom då för att få installera det (paketet). Är det fritt fram eller måste man genomgå någon riskbedömning?
62	FS	Nej, utan då är det fritt fram för utvecklaren att lägga in det, sen så har vi en annan process som kommer ta hand om detta. Som är då vår kvalitets-process/granskningsprocess av kod som kommer in till basen. Där har vi som ett kriterium att man ska tänka på att, okej nu introducerar vi ett tred-jeparts (paket), är licenserna korrekta, får vi använda detta rakt upp och ner? Och där har man liksom ett litet mini-risk (-hanteringsarbete) för den specifika koden. Så att det sköts i så fall av någon annan utvecklare, och behöver man eskalera det gör man det i så fall till en projektledare. Men jag skulle säga att det är sällan man behöver eskalera det, oftast så ser man, man gör en liten riskanalys, man kollar: Vilka använder detta? Vilka sponsrar detta? Vilka är det som har byggt detta och står för underhåll-ningen av det? Och sen kan man också fråga sig: skulle vi kunna ta en snapshot av detta? Och liksom tillåter licensen att vi tar en snapshot, och ser källkoden, så vi bakar in källkoden i vårt eget projekt, för att slippa se att den förändras och sådana bitar, eller kan vi bygga det själv med inspi-ration av samma av samma kod.
63	DS	Nu svarade du på jättemånga följdfrågor [skratt]. Men alltså, risken som man är mest rädd för, är väl helt enkelt att man får in någon form av skad-lig kod? Även om den är försvinnande liten.
64	FS	För oss så är nog risken större, och vi arbetar nog mer med risken, att vi får in för mycket data, och ur en prestandasynpunkt så vill vi adressera detta. Det är nog snarare där som de flesta beslut tas, att nä vi ska inte ta in detta paketet.
65	DS	Ah, alright. Yes. Handlar det också om, jag antar att ni har någon form av löpande underhåll, där det ingår kanske då att man uppdaterar beroenden lite då och då. Blir det arbetet tungt om man har för många (beroenden)?
66	FS	Nä, jag skulle inte säga att vi har massa säkerhetsuppdateringar och pat-char, utan våra projekt med våra kunder brukar vara väldigt långtgående. När vi lanserar siten är vi inte klara, utan vi kommer ha ett partnership med kunden väldigt länge, vi kommer utveckla med kunden väldigt länge och vidareutveckla. Så det blir en del av processen, att ja men, nu ska jag bygga någonting här, och jag ser att den här nya featuren har släppts i det här paketet, då kan vi uppgradera det.
67	DS	För du var inne lite där på att ni tittar mycket på avsändaren av paketet, om de är trovärdiga, och kanske om många andra använder paketet osv? Men ni gör ingen egentlig granskning av paketets faktiska innehåll? Jag förstår att det i många fall är orealistiskt.

68	FS	Jo, men det händer. Men det är återigen, lite av anledningen, vi vill veta vad det är som görs för prestandan av det vi drar in. Jag vet ett relativt nytt fall där vi har dragit in ett animationspaket för att kunna åstadkomma en specifik animation på ett snyggare sätt, och då vet vi hur vi konceptuellt ska lösa det men det tar lite tid så vi kollade på paket, och då kollade vi på källkoden: hur löser dem detta? Det såg bra ut, men paketet hade också en generisk lösning 170 andra animationer som vi inte var intresserade av, och då tyckte vi att det här paketet kommer introducera för mycket kod i våra bundles, så att då tar vi så som de har gjort det och liksom bygger något liknande. Så att det händer, men inte av säkerhets-anledningar.
69	DS	Alright. Men har ni några egna, sådana här lokala, paketregister också? Händer det att ni lyfter in beroenden där så att ni har era egna kopior nersparkade där, eller hämtar ni allt från det offentliga registret när det behövs?
70	FS	Vi har privata
71	DS	Sparar ni kopior från det offentliga i det privata, eller används det bara för egna paket?
72	FS	I vissa fall gör vi det, jag vet inte helt anledningen till urvalet, varför vissa är där och vissa är på public, men jag har för mig att en hel del finns i vårt privata. Jag tror det har att göra med att vi har kunder som har byggservrar som då har brandväggar och inte kommer åt det publika utan vill ha bättre säkerhet, och då har vi lagt in det i vårt privata för att det ska gå därifrån.
73	DS	Yes
74	FS	Men det är en spekulation
75	DS	Ja, vad känner du Hannes, ska vi gå vidare?
76	HC	Ja, jag tror det, jag sitter och kollar lite här, jag tycker att vi har fått bra svar här nu. Men ja, vi kan väl prata... frågan är om vi kanske ska gå vidare till nästa punkt istället. Men ni har ju då ett lokalt register här, det du pratade om innan var ju också att ni har vad du kallade för etablerade paket. Är det då dem som ni har lokalt då, eller har ni liksom någon form av vitlista, hur funkar det?
77	FS	Det jag menar med det var paket som vi byggt själva, från grunden, som vi sparar i vår privata pakethanterare. Och så har vi vissa som även är open source har jag för mig.
78	DS	Alright, men om vi ska gå vidare till nästa punkt då. Om vi tänker oss att nu befinner sig det här systemet, som ni har utvecklat, i någon sorts skarp drift. Det händer ju trots allt då och då att det upptäcks någon vulnerability osv, hur jobbar ni med det löpande?
79	FS	Ja, vi får ju ta en bedömning kring läget, kommer det ut en vulnerability på en lördag, att det här paketet, i den här versionen, det går att på något sätt få in cross site scripting attacker eller liknande, då bedömer vi det. Hur likely tror vi det är att det händer, och är det väldigt likely, ja då fixar vi det på den lördagen liksom. Annars är det liksom... vi behöver fixa det, men det är inget akut läge, då kanske det kommer med i nästa vanliga release. Så det är en prioritering.
80	DS	Hur håller ni er ajour? Använder ni någon typ utav vulnerability tracking-tjänst, eller den som är inbyggt i NPM?

81	FS	Bra fråga, jag vet att vår QA-avdelning har lite sådana, liksom prenumerationer, på olika vulnerability-källor. Men sen så har vi också något inbyggt i vår continuous integration-kedja som kollar, det är säkert mot NPM:s inbyggda tjänst.
82	DS	Ja
83	FS	Jag är inte helt säker
84	HC	Det jag också undrar är... om man tar ett steg tillbaka ur ett bredare perspektiv till riskhanteringsarbete, så då under ett projekt, när det är sjösatt, nu är det här systemet i drift, har ni fortsatt riskhanteringsarbete och hur medveten är man, inte då på managementnivå utan som en vanlig utvecklare? Kommunicerar man om risker? Hur fungerar övervakningen? Hur medveten är man om riskhanteringsarbete?
85	FS	Det är nog väldigt olika beroende på vilket projekt, hur stort projektet är efter lansering, har man liksom lika stort team så fortsätter man bara köra, och då hanterar man då den här riskhanteringen per feature man implementerar. Och det är också upp till projektledaren, vilka risker är det viktigt för utvecklaren att veta och sådana saker. Från erfarenhet skulle jag inte säga att det har hänt, att liksom så, nu måste vi tänka på den här risken, utan i så fall har vi mitigerat den eller så är det en risk som vi vet om och kan inte göra så mycket åt den. Kunden vet om den, vi vet om den, vi liksom har monitoreringsverktyg som ser till att skulle det hända, så är vi snabbt på bollen.
86	HC	Det jag också alltid varit nyfiken på är, hur funkar de här monitoreringsverktygen i utveckling egentligen?
87	FS	Alltså innan vi gått live, eller?
88	HC	Ja egentligen överhuvudet, hur övervakning av risk funkar? Det verkar vara väldigt specifikt för vilken typ av risk, och ibland så är det bara att man tänker på det, och ibland så är det att man har liksom mätinstrument. Jag menar om du jobbar i ett kärnkraftverk så kan du ha ett instrument som kollar på vissa värden. Hur kan det funka hos er?
89	FS	De flesta, nästan all vår övervakning har med stabilitet och prestanda att göra. Så riskövervakning skulle jag säga är något som vårt QA-team jobbar med. Och dem, liksom med dem här prenumerationerna jag snackade om, att de liksom får in data och ansvarar för att hålla liksom ett öra i marken på de paket vi använder. Där finns ju också en risk att det är något projekt någonstans som dragit in ett litet paket som inte finns med på den här radarn. Och den risken existerar, och är förmodligen inte mitigerad idag. Men det är i så fall så det skulle lösas.
90	HC	Ja, vad bra. Och sen till nästa scenario. David?
91	DS	Ja men det är väl lite, för det här med uppdateringar, det är ju lite av ett tveeggat svärd som du var inne på innan. Att visst, man ska hålla sig uppdaterad, men samtidigt så är ju uppdateringar en möjlig attackvektor, att man kan "få in skit" den vägen helt enkelt. Men du var inne på lite att ni uppdaterar inte gärna i onödan? Kör ni t.ex. löst specificerade versionsnummer i er package.json?
92	FS	Vi kör löst specificerade, där jobbar vi med semantic versioning och ser till att paketen vi använder fungerar tillsammans. Men vi har ju också package-lock som då specificerar att det är exakt de här versionerna vi använder just nu, men sen när man uppgraderar kommer den att uppgraderas, men vad som står i package.json är fortfarande liksom en range.

93	DS	Yes. Ja.… det mesta har vi nog fått besvarat annars tror jag
94	HC	Ja, jag tror också det
95	DS	Vi kan nog gå vidare
96	HC	Ja, då har vi i stort sett lite, vad vi här kallar för återkopplingsfrågor. Men det är lite mer generella frågor helt enkelt. Det första vi kan fråga här är väl i stort sett: Hur länge har ni arbetat med NPM? Och har, hur har arbetet med NPM förändrats över tid?
97	FS	Jag skulle gissa, jag gissar att vi arbetat med det kanske 8–10 år, för vi använde... Innan använde vi NPM, inte jättemycket för paket som vi använde på siten, men för byggverktyg och saker för att få bundle-splitting och sådana saker. Och vårt arbete med NPM skulle jag inte säga har förändrats mer än hur NPM utvecklas de senaste 8–10 åren. Vi har inte gjort något aktivt för att förändra vårt arbete med NPM, förutom att vi har blivit ett lite mognare företag som har riskanalyser och sådana bitar. Men inget anmärkningsvärt.
98	HC	Ja, och då kan vi också, det sitter ihop lite här, vilka fördelar finns med NPM? Eller kanske ännu viktigare, vad hade hänt om man inte skulle använda NPM? Vad hade det inneburit och vad gjorde ni tidigare?
99	FS	Tidigare var inte paketdelning och open source en lika stor grej. Det innebär ju att vill du ha någonting får du bygga det själv, och det i sig leder till att det blir fler buggar och förmodligen fler liksom säkerhetsluckor, för att man måste bygga allting från scratch, det blir inte perfekt första gången och det blir bara reviewat av kanske en eller två utvecklare på det företaget. Så att det innebar mer risk, ur det perspektivet, och även mycket långsamma projektimplementationer. Så att fördelen med NPM för oss är ju att vi kan, vi kan använda befintliga bibliotek eller paket eller liknande för att lösa dem problemen vi har på ett väldigt snyggt sätt som gör det bra för slutkunden. Ett exempel på det är att vi bygger saker i React, men React är ju också ett paket från NPM från Facebook, och vi hade nog aldrig haft kapaciteten att bygga ett liknande bra ramverk som är lika stabilt, lika prestandaeffektivt och sådana bitar, så att om vi inte hade haft NPM så hade vi byggt en sämre lösning som inte var lika vass, som var jobbigare att utveckla och alla sådana bitar.
100	HC	Så att på det stora hela så innebär det egentligen att använda NPM är mer säkert än att inte gör det, i stort sätt? Kanske inte säkert, men att man exponerar sig för mindre risk än vad man gjorde tidigare?
101	FS	Det beror ju lite på scenariot, men precis, allting är en trade-off. Trade-offen är att du byter risken mot en annan risk. Nu finns det ju sätt att du får in kod i din kodbas som är malicious, och det är en risk, men å andra sidan så finns risken att du bygger vulnerabilities i koden utan att veta det för att du inte har samma granskningsprocess som ett open source-bibliotek har, så du byter en risk där. Jag kan inte svara för [***] men jag skulle personligen misstänka att vi värderar risken större att vi introducerar saker själv om vi bygger så mycket från scratch, än att vi tar etablerade paket från etablerade företag.

102	HC	Ja, och det för mig också in på en annan fråga, jag tror vi pratade lite om det tidigare, men det här, hur pass villig man är att ta en risk när vi pratar om prioriteringar, att man gör risk-scores genom en matris, och så spottar den ut en automatiserad prioriterad lista. Och då är det ju såklart, hur stor sannolikhet det är att det inträffar mot då konsekvensen. Men det kan ju då finnas tillfällen, precis som du sade, att det finns då en risk som har ett förhållandevis högt "score", men där man ändå säger att detta är värt att vi utsätter oss för. Och hur gör man då? Hur går resonemanget till egentligen? Är det vissa saker som inte går att komma ifrån, men där man ser att vinsten ändå är så pass stor att det är värt att ta den (risken)?
103	FS	Ja, det är precis som du säger, det handlar om vinsten, dvs trade-offs. Är det en stor grej som påverkar projektets estimat eller utvecklarens tid att lägga på det så kan det vara en trade-off som är värd att ta en risk för. Och är det en väldigt stor grej där vi ser att risken är rätt stor, nu kan jag inte komma på något exempel på det för det låter lite fiktivt, men i så fall skulle man då eskalera det till kunden och säga att: vi kan sänka kostanden för det här projektet med så här många timmar och därför pengar, men det kommer också innebära att vi kanske använder det här tillvägagångssättet, som har den här trade-offen. Så det är väldigt baserat på vilken situation man sitter i, men man eskalerar det och tar beslut utifrån det.
104	HC	Det är spännande, för vi har läst ganska mycket om vissa riskhanteringsramverk där man har en algoritm som spottar ut scores, och sen beroende på hur de är prioriterade så är det man kör på i stort sett. Och får dem någon form av tröskelvärde så "skiter man i det".
105	FS	Just det, vi har inte så stora etablerade... det är fortfarande väldigt personliga riskprocesser där vi sitter med människor från olika delar av projektet och kvalitetsansvarig och sådana, så att allting som tas upp där faktiskt diskuteras. Där är ju trade-offen att man kanske blir mer pragmatisk och lite mer erfarenhetsbaserat kan se på riskerna. Men det är också dyrare för oss att har så många människor i en workshop, den (kostanden) är vi idag villiga att ta.
106	HC	Jag menar att teorin sällan stämmer överens med verkligheten. Det märks ju att ni har en rätt så tydlig strategi gällande riskhantering, är det något du kan liksom, se en linje mellan hur ni väljer att hantera NPM-paket och hur er riskhanteringsstrategi fungerar till liksom er övergripande företagsstrategi, känner du att de hänger ihop?
107	FS	Jag skulle inte säga att jag kan se några direkta kopplingar mellan dem utan det är någonting som har växt ur erfarenhet. Vi har sett tidigt, för många år sedan, att för varje incident man har så har vi en incidenthantering, eller ibland har vi inte en incidenthantering, men utvecklarna lärde sig något och liksom att man tar med sig det. Och det har lett till att vi rätt naturligt går in att vi vill ha en riskprocess rätt tidigt i ett projekt. För att vi vill lära oss av våra misstag och vi vill liksom hantera, vi vill vara inte reaktiva utan proaktiva på saker som kan hända, vi måste snacka med kunden redan nu. Så jag skulle säga att det är något som växt organiskt tillsammans med vissa individer och roller som har gjort att vi blivit väldigt mycket bättre på detta arbetet. Men det är inget som är implementerat on the fly utan det har vuxit hela tiden och vi blir bättre på det hela tiden.
108	HC	Ja, har du några ytterligare frågor David?
109	DS	Nej, jag känner mig rätt nöjd

110	HC	Ja, för att jag tänkte lite på, det har vi redan pratat om i och för sig, men det här som du och jag David har pratat mycket om, det tveeggade svärdet med att antingen strunta i uppdateringar eller att bara acceptera dem, och att man hamnar i någon form av moment 22 där, alltså att oavsett hur man gör så utsätter man sig för olika typer av risker? Som någon form technical lag eller att man får in "skit". Men ni kör i stort sett alltid på att installera så få uppdateringar som möjligt?
111	FS	Nej det skulle jag inte säga, utan jag skulle säga att vår strategi är snarare tvärtom, att vi kör uppdateringar av större paket för att hålla oss up to date. Det är en kombination av att vi inte ska introducera technical debt, dvs att ska vi uppdatera dessa paketet måste vi också uppdatera dessa 50, och då blir det 200 timmar i arbete, men också för att som du säger att, att ha gamla paket innebär ju inte att de är 100% safe utan man kan hitta vulnerabilities som är patchade i nya versioner. Så där skulle jag säga att vi försöker hålla oss... på de stora paketen så håller vi oss rätt mycket i framkant, men det ska också finnas någon form av "reasoning behind" varför man uppdaterar. Det är inte som att vi bara "det här lilla animationsbiblioteket uppdaterar vi så fort det kommer ny release", nej, det är liksom, det är inget "breaking", det fungerar som det ska, det här biblioteket är inte kopplat till något annat heller så det lever rätt isolerat, så då kan vi köra någon "update-push" någon gång, men större paket som arbetar mycket mer tillsammans, de ser vi till att liksom hålla färska, framförallt att kolla på releaseloggar och se vad som kommit nytt och bedöma utifrån dem. "Det här nya stora paketet vi använder har fått en buggfix för en sak vi vill ha" - Great! då hoppar vi på den bollen och uppdaterar, och även uppdaterar det som är relevant till det paketet så att vi inte sätter oss i någon skuld inför framtiden.
112	HC	Då låter det ju onekligen som att ni också har någon nivå av granskning utav en uppdatering, att ni inte bara accepterar den, utan att den ska uppfylla någon form av krav, antingen att den ska vara nödvändig eller...
113	FS	Precis, och det här leder nog egentligen snarare till att våra kunder ska betala oss för vårt arbete, och då ska kunderna också gå med på att vi göra någonting som inte ger direkt värde. Hur ser vår ROI ut på de här 10 timmarna du suttit och uppgraderat saker, vad får vi för värde av det? Så att där behöver vi vara motiverande och förklarande, många av våra kunder är ju väldigt etablerade företag som har koll på läget, så de kommer inte med sånna dumma frågor, men som en princip så behöver vi ha en motivering. Vi kan inte bara sitta och göra vad vi vill.
114	HC	Ja precis, ja, tack så mycket. har du några fler frågor David?
115	DS	Nej, det har jag inte
116	HC	Ja, men då ska vi väl ta och tacka för oss
117	FS	Tack själv!
118	HC	Stort tack, det här var väldigt bra, vi fick jättebra svar.
119	FS	Härligt
120	DS	Tack så jättemycket!

Appendix B

Transskript Produktutvecklingsbolaget.
30/4–2020, via Google Meets.
Längd: 45 minuter

Deltagare och förkortningar:
Författare: David Strömbäck (**DS**), Hannes Carlsson (**HC**)
Respondent: Medgrundare på Produktutvecklingsbolaget (**MG**)

Rad	Talare	Tal
1	HC	Önskar du och företaget anonymiseras?
2	MG	Det behöver inte vara anonymt.
3	HC	Hur skulle du beskriva verksamheten?
4	MG	Det är primärt ett konsultföretag. Där vi jobbar med alla typer av olika kli-enter, med allt som har med digital design att göra. Vi har också vårt eget verktyg som vi bygger, som vi använder själva för att bygga prototyper och andra typer av digitala upplevelser, men det licensierar vi även ut till andra. 97 % konsultföretag och 3% produktbolag.
5	HC	Är systemutveckling en kärnverksamhet?
6	MG	Ja, det skulle jag säga. Det är en stor del av vad som gör oss annorlunda, att vi har det här verktyget som vi kommer med till kunderna som får oss att sticka ut. Det är ett sätt att differentiera oss.
7	HC	Vilken roll har du på företaget?
8	MG	Jag är en av [***] medgrundare, och jag sitter i kundprojekt, men även med att utveckla vårt prototypverktyg.
9	HC	Hur stora är ni?
10	MG	Runt [***] pers.
11	HC	Hur stor omsättning har ni?
12	MG	Runt [***] miljoner.
13	HC	Vilka kunder har ni?
14	MG	Ganska blandat, jobbar ganska internationellt, en del stora företag, tex [***], [***], [***], jobbar lite med startups också, ett företag som gör [***] i [***], [***] lokalt, har jobbat med [***], ganska spritt.
15	HC	Arbetar ni agilt?
16	MG	Nej det skulle jag inte säga, lite skillnad från vad vi gör.
17	HC	Använder ni NPM-paket i er utveckling?
18	MG	Ja, det gör vi. Väldigt många!
19	HC	Ett scenario: ni ska dra igång ett nytt utvecklingsprojekt, hur ser riskhante-ringsarbetet ut i ett inledande skede? Vilka riskhanteringsaktiviteter finns? Mer brett än NPM, för all typ av risk, finns det planering, identifiering, ana-lys, prioritering?

20	MG	Menar du för teknisk risk?
21	HC	Nej, för vilken risk som helst.
22	MG	Nej, vi har väldigt lite formellt sådant. Inget alls. Det är helt baserat på individer skulle jag vilja påstå, framförallt då vi är så få som sitter med det, så jobbar vi mer som en startup, försöka ta de få timmarna man har och försöka göra de så produktiva som möjligt, och det gör ju att man nästan alltid prioriterar ned kringaktiviteter, saker som får kunder att vilja ha det, att göra bra marketing, att göra mycket business development, sen även utveckla de features som behövs, det är ju prio ett, och sen saker som kodkvalitet, riskhantering, processer runt det, även till och med saker som att få ett bra deploy workflow och [ohörbart] och sånt där det hamnar på efterkälken, eftersom allt fokus hamnar på att försöka få någonting som någon vill ha, för att bygga någonting som ingen vill ha så finns ingen poäng med att bygga det. I det tidiga skedet blir det mest att försöka hitta någonting som kunderna vill ha och efterfrågar, sen försöka testa det så snabbt som möjligt. Mer som kanske experiment, kanske inte tänka att nu ska vi bygga något färdigt, utan nu ska vi bygga det minsta möjligt för att få feedback, för att se om vi är på rätt väg. Vi är väl oftast i det skedet, det är ganska sällan det slutar med - nu vet vi vad kunden vill, nu är vi redo att bygga den första versionen. För det är oftast våra kunder som gör det! Vi gör ju det här andra arbetet åt kunderna. Men sen även själva med vårt eget prototypverktyg, så är väl riskerna, åtminstone de tekniska, ganska små eftersom vi säljer ofta direkt till kunderna, vi har inte så mycket organisk trafik vi hemsidan som får folk att ladda ner det, utan det är mer baserat på vår konsultverksamhet så kan vi komma med vårt verktyg också, så vi har ganska bra kontroll över vilka kunder vi har och vilka som har det, och hur det distribueras och så vidare. Det kan vi göra ganska manuellt, tack vare att det är så småskaligt. Hade det varit mycket större så hade det varit... Vi har redan börjat komma till att vi behöver börja implementera bättre processer. Men vi väntar så länge vi vågar, kan man väl säga. Sen kommer det en punkt när man inte vågar vänta längre med riskhantering, för när folk väl börjar använda det och betala för det, och det börjar bli seriöst, då är det värt att investera tiden i att få mindre risk. Det värsta som kan hända är att om man säljer någonting och så förstör man det via någon uppdatering som går snett, eller någonting som får kunderna att inte kunna arbeta med det, då kommer de bara bli arga och slänga det och gå och göra någonting annat. Men innan det händer, så är det så lite som möjligt beyond att ha bra lösenord på sitt AWS-account ungefär.
23	HC	Så kan det ofta se ut, att det finns inget strukturerat riskhanteringsarbete men det sker ändå implicit. Ni har kanske inte riktlinjer och processer för hur allting ska gå till, men det kanske ändå inte är vilda västern där alla får göra precis hur de vill? Finns det på individnivå ett tänk och kommunikation om riskhantering?

24	MG	Vi litar på de personerna som är där att inte göra någonting för dumt. Så saker som att, vi har ju kunderna som använder vårt verktyg, de bygger ju designer som är konfidentiella, som absolut inte får kunna spridas eller visas utav andra, och det är ju väldigt stora problem - om vårt verktyg skulle sprida något som [***] gör tex, då skulle de ju stämma oss tills vi inte finns längre. Sen har vi flera kunder, och en backend, så skulle en kund få tillgång till en annan kunds design så skulle det vara slutet för oss. Där finns ju ett säkerhetstänk baserat på att inte läcka information mellan kunderna, men än en gång inte formellt eftersom vi är så pass få som gör det utan mer att vi litar på varandra, och vi vet att folk har byggt grejer innan och har lite koll på det, så väldigt icke formaliserat, men undermedvetet nästan.
25	HC	Och där spelar väl erfarenhet väldigt stor roll kan jag tänka mig?
26	MG	Ja, det gör det verkligen. Man behöver ha lite koll på koncepten och veta vad man behöver tänka på. Och kunna använda rätt funktioner på AWS tex, för att se till att segregera data så att en användare inte kan läsa en annans grejer. Och det handlar mycket om att försöka använda deras grejer och inte göra våra egna, försöka följa best practices, det blir enklare och enklare tack vare att de paketerar dem åt en. Tex all användarhantering gör vi via AWS Cognito, och då får man otroligt mycket på köpet, när det gäller just säkerhet i alla fall.
27	HC	Så då fattas en del beslut gällande arkitektur och tredjepartskomponenter i planeringsfasen redan?
28	MG	Ja, planeringsfasen är ganska kort, det är mer att vi har en idé om vad som behövs, då sitter vi inte och planerar så länge utan mer att, vi tror att, det finns tydliga tecken på att det finns kunder som vill ha det, då bara försöker vi bygga det så snabbt som möjligt för kunderna att verifiera att så är fallet. Och sen nästan efter det kanske, OK det här är något folk vill ha, hur ska vi göra det på ett bra sätt, det är mer på det hållet. Man kan ju kalla det experimentet för en del av planeringsfasen om man vill. Det är mer att minimera risken för att spendera onödig tid. Det är egentligen det värsta som kan hända, om vi bygger någonting väldigt väl, men ingen använder, det är en otroligt stor risk för oss att ta. Så vi vänder hellre på det, där vi bygger det först, och sen tar bort de tekniska riskerna i efterhand. Sen så givetvis med Cognito får du både och, du kan göra det snabbt och ganska säkert, på samma gång, det finns inte så mycket trade-off mellan tid och säkerhet där.
29	DS	Så kan man i princip säga att ni sätter igång och börjar utveckla ganska omgående? Är det så då att varje enskild utvecklare får installera paket och beroenden som de anser sig behöva och kan underlätta arbetet?
30	MG	Ja
31	DS	Och där finns ingen process i det skedet, utan den kommer senare, att man utvärderar alla beroenden man har tagit på sig?
32	MG	Ja. Det är inte så att vi har en formell, nu ska vi utvärdera beroenden, eller har kommit till en punkt där vi behöver kolla på det, utan det är mer under, eller vi ska säga att när man väl installerar ett NPM-paket så brukar det finnas en viss minimi nivå, av att kolla licensen, se att den är fine, och sen kanske tänka efter lite grann att se att den inte kör några konstiga run scripts vid installation och sådär.
33	DS	Precis, då gör ni ju ändå någon sorts granskning. Ni kanske tittar på avsändaren också? Är det en paket från en anonym person eller är det ett paket från Facebook tex.

34	MG	Precis, men nu är vi mer inne på individnivå, nu pratar vi mer om vad jag gör. Men vi kan ta det, jag tror att de flesta gör ungefär liknande, och det är väldigt basic. Det är att kolla - är det många contributors eller många commits, har de funnits länge, gör de det, och de har MIT-licens, då brukar jag känna att de är fine.
35	DS	Händer det att ni kikar på källkoden och kollar vad som gömmer sig i paketet?
36	MG	Inte för säkerhets skull, utan det är i så fall av andra anledningar, mer funktioner.
37	DS	Upprättar ni någon dokumentation i samband med att ni tar på er ett nytt beroende, som bill of materials?
38	MG	Inte mer än package.json.
39	DS	Nej den blir ju ganska självdokumenterande.
40	MG	Och lock-filen, men den kollar jag inte på.
41	DS	Har ni något privat register? Det finns många verksamheter som sätter upp egna register, både med egna paket och med kopior av offentliga paket, är det något ni jobbar med?
42	MG	Nej, det skulle nog komma ganska sent för oss, alltså det skulle nog aldrig komma nu med den storleken vi har, för den investeringen både pengar och tid, kommer nog aldrig betala tillbaka sig. Jag tror att ur riskanalys är det bättre att ta risken, än att faktiskt sätta upp det här privata och investera den här tiden och resurserna för att underhålla det. Och även att, varje gång man ska installera ett NPM-paket blir det lite mer process runt det. Och jag tror faktiskt att det är nästan en större risk för oss, att gå långsammare, än säkerheten det ger oss. Men hade vi varit 100 personer hade det varit mycket enklare. En given dag är vi kanske tre personer som sitter och utvecklar någonting, som används på riktigt, av tre personer, den overheaden att ha ett privat register som man maintainar, det blir en ganska hög procent. Men om man är 20-30-40 personer, då hade det varit en mycket lägre impact på vår produktivitet.
43	DS	Om vi går vidare i det här scenariot, och vi tänker oss att ni nu har ett system som befinner sig i skarp drift, hur jobbar ni då med uppdateringar av NPM-paket? Det kommer ju security advisories har ni någon som ansvarar för att hålla koll på dem och åtgärda dem med uppdateringar?

44	MG	Nej, det har vi inte. En anledning till det är att de flesta security eller audit-grejerna, om du kör kanske en hemsida som är publik online, kan du få vissa bekymmer med cross-scripting injection, databasproblem och sådär, eller om du kanske kör främmande Javascript i ditt environment på något vis, finns det mycket luckor. Men vårt verktyg är en Electron desktop-app, som kör projekt du skapar från lokala filsystemet eller vårt eget cloud-backend som är helt privat och gömt, eller så kör du på din telefon, det som vårt prototyp-verktyg producerar som är en React-sida, men den har ju bara det som är exakt det vårt verktyg spottar ur sig, som då är ganska self-contained, det kommer ju inte komma in någon främmande kod efter det. Attackytan är väldigt liten. Så jag har nog aldrig sett ett NPM-audit meddelande som jag har känt mig orolig över i kontexten av vår app, så om man får ett meddelande om att det finns ett säkerhetsproblem med Mongo DB med nycklarna, i det här eller det här fallet, ja det gör ingenting för att det här är en lokal instans som körs på varje kunds dator som är helt stängd, det finns inget sätt utåt att komma åt det. Så det förenklar för oss ganska mycket i det fallet, att det är ganska mycket som är lokalt, väldigt lite publikt öppet attackyta.
45	DS	Då låter det som att ni kanske inte går in och bumpar versionen på NPM-paket i onödan heller?
46	MG	Nej, det är väldigt mycket tvärtom, att varje gång vi gör det så blir det alltid problem, compatibility-problem med saker som inte funkar som det ska, och tack vare, eller på grund av Javascript så är det ofta ganska svårt att hitta dem om man inte... [tekniskt fel]
47	DS	Nu försvann ljudet för mig här.
48	MG	Ja det hackar lite. Det är inte som i native-språk där det är att kompilerar det så hade det antagligen varit OK, utan det är ofta lite mer vilda västern i Javascript, även många NPM-paket ändrar API:s TROTS att det bara är en patch.
49	DS	Precis, det slarvas väldigt mycket med det där. Men använder ni semantic versioning när ni specificerar era versionsnummer? Man kan ju specificera att jag vill ta emot patch och minor eller vad man nu vill.
50	MG	Jag tror att som det ser ut nu, de stora paketen som webpack och babel och såna, där låser man sig till minor, och de mer privata, lite mindre grejerna, där kanske man vill låsa sig till exakt en viss version. Men det är oftast retroaktivt när man har råkat ut för ett problem, så låser man det. Tex att någonting uppdateras, så går det sönder, och då fryser man till den gamla versionen. Och sen ligger den där fryst för evigt [skratt]. Det är väldigt ostabilt skulle jag säga, NPM-paket i allmänhet. Det är en så otrolig skillnad i kvalitet mellan dem, så man vågar inte uppdatera dem i onödan.
51	DS	Det här med uppdateringar har vi pratat lite om, men det är lite av ett tveeggat svärd, samtidigt som man såklart ska hålla sig uppdaterad för att minimera säkerhetsrisker, så är uppdateringar också en attackvektor. Och det har vi sett i ganska många fall att man har fått in skadlig kod som är förklädd som en säkerhetsuppdatering till exempel har ni något resonemang kring det?

52	MG	Jag skulle nog säga att vi tar nog relativt stor risk där. Och hoppas på det bästa [skratt]. Det skulle ju lätt kunna tänka sig att ett av våra beroenden tas över av någon annan som lägger in något virus istället för en uppdatering, och det kommer in i vår app som vi sen vi skickar ut till kunder. Och det är ju Electron så att du kan ju få in native C-komponenter och så vidare. Det finns ju ingen begränsning egentligen med vilken typ av malware du skulle kunna lägga in i vårt verktyg. Och sen så distribuerar vi det. Det finns absolut en risk att det skulle kunna hända. Den är ju, tack vare att vi nästan aldrig uppdaterar paket [skratt], och vi har nästan bar kända välanvända paket så är väl risken relativt låg. Men det finns ju inget tekniskt som stoppar någon från att kunna göra det. Jag vet inte hur bra processer webpack och babel och de där har, men vem vet om någon skulle kunna committa in någonting dumt där.
53	DS	Ja, problemet är väl att de i sig har 70 nya beroenden som man kanske inte har stenkoll på.
54	MG	De verkar ju inte heller ha så bra riskhantering alltid, även de stora.
55	DS	Ja, det är ju lite intressant, vem det är som egentligen äger problemet, och var i kedjan man ska sätta sin tillit?
56	MG	Ja egentligen så, man kan ju inte begära någonting när det är gratis och en licens som säger "gör vad du vill", att begära något av någon annan där. Jag tycker väl att det hamnar på oss, när vi väl tar betalt för någonting, då är ansvaret på oss, och det är ju där det hamnar, skulle det bli problem då är det ju vi som blir stämda. Så rent legalt så är ansvaret på oss också.
57	DS	Skulle det inträffa, det här scenariot du beskrev precis, då är ju konsekvenserna ganska allvarliga, men samtidigt låter det som att ni bedömer att chansen för att något så allvarligt inträffar är försvinnande liten?
58	MG	Ja, det kan man väl säga.
59	DS	För det är ju alltid en sån kalkyl, konsekvens vs. hur stor chans det är att något sånt inträffar.
60	MG	Exakt. Jag tror att lösa problemet är en större risk än att ha problemet. Men hade vi varit... [tekniskt fel]. Oj nu är vi tillbaka. Hade vi varit en startup med miljoner i investering, då hade man nog ökat riskanalys lite jämfört med nu, där det inte är lika kritiskt.
61	HC	Du pratar om att de paketen ni använder är stora och att ni kan ha någon form av tillit till dem, då låter det som att ni gör någon form av avvägning någonstans över vilka paket som ska inkluderas, hur funkar den avvägningen, tittar man då på avsändaren och populariteten, är det upp till varje individ eller har man en kommunikation sinsemellan om den här typen av val och risker?
62	MG	Primärt helt per individ, men man ser vad andra lägger in, så det finns en form av icke formell code-review, men den händer, jag ser ju om jag drar in något nytt från Git, att oj nu har något ändrats i package.json, jag måste göra en NPM- install, det är svårt att undvika att det har lagts in någonting och då kan jag ju se det. Sen om jag agerar på det är en annan sak, men en viss sekundär koll blir det. Men än så länge har vi klarat oss undan. Vi har inte så fruktansvärt många dependencies där tex om det är någonting litet, alla utvecklare har en tendens att bygga det själva. Vi har inga left-pads och såna grejer [skratt].
63	DS	[skratt] Nej just det! den är legendarisk.

64	MG	Ja exakt. Nej det är mer att vi har mycket som vi byggt själva som skulle kunna vara dependencies, det är jag rätt säker på, men vi undviker ju risk och faktiskt även ofta tid genom att bara bygga det själva, förutsatt att det inte är något allt för stort. Det gäller de flesta utvecklarna. Så vi har en tendens att bara ha stora välkända dependencies, förutom vissa små, väldigt nischade grejer, där det mer är kanske ett repo som är fyra år gammalt, vi har några såna. Men det är inte alls så många.
65	HC	Men kan det då ske att ni är intresserade av ett paket som ska utföra en specifik mindre funktion, och så tittar ni då på det här paketet och ser att den gör så mycket annat och är fylld av grejer vi inte är intresserade av, kan man då ta den funktionaliteten bara, bygga den själv? Finns det fall då man tittar på paket och bestämmer att nej inte det, men vi gör nästan samma sak fast själva?
66	MG	Ja det stämmer. Är det för mycket, och vi bara vill ha något litet, vi har nog inga såna tror jag, beroende på hur man ser webpack och babel och sånt där, man använde sällan en stor del av det. Annars är det oftast paket som gör exakt det vi vill göra. Tex interagera med AWS S3, på något specifikt sätt, om man vill synka en mapp lokalt till S3, så finns det alltid någon som gjort ett sånt paket som är 200 rader Javascript, som är exakt det, det har vi en handfull av. Och sen resten är verkligen stora saker som gör exakt det vi vill göra. Försöker undvika saker där vi bara använder en liten del, ofta är det ganska komplext, sen har den andra dependencies som ställer till det för andra paket och det blir så mycket bekymmer runt omkring, bara för att få den här lilla lilla biten man vill ha. Att det bara är värt det att göra det själv.
67	HC	När ni utvärderar paket, är prestanda då en av de saker som ses över? Att ett paket kan göra flera saker än just det som ni behöver?
68	MG	Ja, det kanske inte har hänt, för oftast är det att det ligger död kod, men oftast inte att något faktiskt kör, men prestanda är en sån grej att vi profilerar så att det fungerar som vi vill och att allt flyter på, det är inte förrän testerna flaggar något som vi faktiskt tittar på det. Så länge de är glada så är vi glada.
69	HC	Då går vi vidare från de här scenarion till övergripande återkopplingsfrågor. Hur länge har ni arbetat med NPM, och hur har det arbetet förändrats över tid?
70	MG	Vi arbetar med NPM på två sätt. Det första är i konsultdelen där vi jobbar direkt med kunder, där vi bygger en designprototyp kan man kalla det, inte en produktionsprototyp utan någonting för att bevisa designen. Säg att det är ett hemlarm där man vill testa röst input och output så vill man att saker och ting ska prata till höger och vänster, och interagera på alla möjliga olika sätt med olika enheter, och då måste man i princip bygga någonting för att kunna avgöra och testa med riktiga användare om det är bra eller dåligt, vi bygger mycket sånt. Men det är bara en design, så när det väl har bevisats så tar produktionsavdelningen hos kunden över och bygger det själva, där våra grejer mer är en designspecifikation än en teknisk. Där kan vi installera vilka NPM-paket vi känner för, till höger och vänster, hej vilt, för vi vet att om en månad så kommer det här aldrig röras igen.
71	HC	Så det är mer proof of concept?

72	MG	Ja exakt, och du kör bara på enheter vi installerar det på, där har vi ju verkligen bara hastighet i åtanke, ingenting annat, och det har ju aldrig varit ett problem och inte ändrats, så har vi jobbat med det i sju år. Och sen är vårt prototypverktyg där började det så, tills vi fick kunder som betalade för det, och då är det ju jättestora internationella företag, så man blir lite nervös, och då ändras det till att vi gick igenom och kollade så att är alla lösenord OK, är AWS-backenden säker nog, och är den säker nog då kan det ju komma in vilka hemskheter som helst i klientappen, de kommer ändå inte kunna göra så mycket, i princip. Men klientappen är ändå så att vi prioriterar hastighet framåt mer än säkerhet fortfarande. Och det ser inte jag ändras tills vi har ännu fler kunder som använder det, där risken ännu större, för nu är risken att vi inte kommer få tillräckligt många kunder så det inte är lönt att fortsätta. När det är risken så är det bara att komma så snabbt framåt som möjligt som premieras. Så en brytpunkt när vi fick en betalande kund, det får jag ändå säga.
73	HC	Hur ser ni på fördelarna med att använda NPM? Vad hade konsekvenserna varit om det inte hade använts?
74	MG	Jag tycker det gör väldigt stor skillnad i produktivitet, tex om man sitter i C++ där det inte finns något sådant, nu har ju alla nya tekniker någon form av pakethanteringssystem, även native som rust, och det finns väl tredjeparts till Apple osv. Jämfört med så som det var innan är det massiv skillnad hur snabbt man kan röra sig framåt med bara några få personer. Innan satt jag i evighet och bara konfigurerade byggsystem för alla dependencies, tog ju hur lång tid som helst ...[tekniskt fel]... men nu är det ju bara NPM- install och så har du det du behöver, det är en massiv skillnad. Det är också lätt att testa, man kan testa tio olika paket och se vad som funkar, förr var det en overhead som var helt löjlig, det var inte lönt. Så jag tycker det har ändrat väldigt mycket, hur snabbt framåt man kan komma, och även hur roligt det är att programmera, det är mycket roligare att kunna dra in vad som helst och köra, man behöver inte sitta och spendera mycket tid på smådetaljer som man egentligen inte vill lösa.
75	HC	Säg att din kollega skulle skriva någon funktionalitet, och så finns det samma funktionalitet i ett paket med en miljon nedladdningar, vilket bedömer man då är säkrast?
76	MG	Jag skulle nog säga att om det inte tog väldigt lång tid att bygga själv så hade jag föredragit att bygga det själv. Vi får alltid någon gång under utvecklingen problem av att paket har uppdaterats. Eller vi uppdaterar Electron och sen blir någon gammal dependency inkompatibel, och alla de grejerna är så himla tråkiga så jag skulle bygga själv om det är möjligt är nästan alltid bättre, men det finns en tydlig gräns där det inte är värt det. Du bygger inte gärna webpack själv. Och även i det fallet med att synka en lokal mapp med S3 där det är 200 rader kod, den har väldigt låg risk och har varit stabil i tre år, det finns inte så mycket där som kan ställa till det. Skriva själv är nog att föredra, förutsatt att det är praktiskt.

77	HC	Vi har rört vid det ett par gånger här, med hur pass villig man är att ta en risk. Hur stor chansen är att något inträffar kontra konsekvensen av ett sådant inträffande kan ju ge en typ av score på en risk. Men man kan inte bara spotta in det i en algoritm som ger en massa riskvärden, där vissa risker har väldigt höga värden så dem borde ni gå runt, riktigt så funkar det kanske inte? Hur gör man den avvägningen om vilka risker som är värda att ta, och vilka som inte är värda att utsätta sig för?
78	MG	Det var en väldigt bra fråga. Jag tror att i vårt fall så görs nog inte den bedömningen explicit, utan vi har mindsetet att vi vill komma framåt så snabbt som möjligt och verifiera att det vi har är något som går att sälja, och tjäna pengar på, så vi kan lägga mer resurser på det. Och fram tills att vi känner oss bekväma med att lägga mer resurser på det så är det bara... Risken vi känner när vi sitter med det nu är att det inte kommer bli något av det, och den trumfar egentligen allt annat som får oss att gå långsammare, och vi har jobbat med många kunder som är i tidig fas med att bygga något nytt, alltså något helt nytt inte en iteration, och då är det nästan alltid så att det organisationen är mest rädda för är att det de bygger inte kommer bli någonting. Riskanalysen, framförallt den tekniska och säkerhetsbiten, den hamnar ju långt ner på listan, tills det blir problem som att en kund kan se en annan kunds data, så det ser vi till att det funkar. Men det är allt.
79	HC	Men då låter det som att ni har identifierat den största risken att man tar för lång tid på sig, eller att det man gör inte går.
80	MG	Ja precis. har vi en idé om något vi vill bygga innan det har visat sig att det var en bra idé, då är varje timme investerat i den potentiellt bara ren förlust. Så man vill ju göra två saker rätt, man vill spendera så få timmar som möjligt tills det blir tydligt att det är värt att fortsätta. Det kan ju vara att man har tio idéer, då vill du kanske göra tio experiment. Om det bara är en landing page på en grej, kanske bara några intervjuer på andra grejen, det kanske behöver byggas lite demos på den tredje grejen och så vidare, men man vill ändå göra det minimala för att se vilka grejer man går vidare med. Och sen när man går vidare så kanske man bygger lite mer demo, så lite som möjligt, och sen långsamt långsamt till man känner att det här kommer bli någonting.
81	HC	Då låter det som att hur ni väljer att arbeta med risk är starkt knutet till er övergripande strategi egentligen, det är såhär vi arbetar, därför gör vi de här valen i stort sett?

82	MG	Ja. Där är också hastighet en viktig faktor, både för oss, men även, vi jobbar mycket i stora innovationsprojekt, där etablerade företag vill komma på nya idéer, nya produkter, och där är ju verkligen tiden bland det mest kritiska. Både för att organisationen inte ska tröttna på idéerna, så att Vd:n inte kommer och tycker att nej nu går det här för långsamt, nu får ni göra någonting annat, det är en risk man vill ta bort, eller att nej det här kostar för mycket, eller att det är svårt att få loss ingenjörsresurser, och det hetaste just nu är att arbeta Lean, spendera så lite resurser som möjligt och så lite tid som möjligt för att nå maximalt resultat. Om organisationer har det tänket kommer riskhantering på NPM-nivå väldigt långt ner på listan, det finns ju stora kunder som verkligen blir bitna av det, där det blir ett problem. Det är förvånansvärt lite riskhantering i många företag stort som litet, möjligtvis när det är väldigt tekniska företag, så har de lite bättre koll. Deras rykte står på spel också. Men många som gör konsumentgrejer de lägger det långt ner på listan, även tills det blir ett problem, där de borde ha det högre upp.
83	HC	Då tror jag inte vi har några fler följdfrågor, eller jag känner mig rätt nöjd.
84	DS	Ja jag är också rätt nöjd.
85	MG	Vilka företag har ni pratat med?
86	DS	Vi pratade med ett större företag igår.
87	HC	Sen är det ännu mindre företag också. Vi vill prata med företag ur lite olika storlekar, för att se lite om riskhantering här hänger ihop med storlek, och det bekräftar ju vårt samtal idag lite, som du säger att hade ni varit 100 pers så hade det nog sett annorlunda ut.
88	MG	Ja, det hade det. Vi är ju 20 designers kanske, som inte sitter med NPM över huvud taget, och kanske 5–10 personer som gör det, så vi är inte 30 utvecklare heller.
89	HC	Tusen tack för att du tog dig tid!
90	MG	Ingen fara!
91	DS	Tack så jättemycket!
92	MG	Ingen fara! Lycka till

Appendix C

Transskript AI-konsultbolaget.
Videosamtal, 6/5–2020, via Zoom.
Längd: 45 minuter

Deltagare och förkortningar:
Författare: David Strömbäck (**DS**), Hannes Carlsson (**HC**)
Respondent: Partner och tekniskt ansvarig på AI-konsultbolaget (**TA**)

Rad	Talare	Tal
1	DS	Då kan jag börja med att ställa lite inledande frågor här. Önskar du vara anonym, och önskar företaget vara anonymt?
2	TA	Det har jag faktiskt inte kollat upp, men det kan vi nog säga att det behöver det nog inte vara.
3	DS	Du kan ju bara höra av dig om ni kommer fram till något annat.
4	TA	Ja, absolut.
5	DS	Hur skulle du beskriva företagets verksamhet?
6	TA	Man kan väl säga att vi är ett relativt nystartat konsultbolag, vi har funnits i ett och ett halvt år kanske, vi fokuserar på data engineering, data science, machine learning, liknande projekt. Men vi har även en del full-stack-jobb, webb, och en app nu senast.
7	DS	Skulle du säga att systemutveckling tillhör eran kärnverksamhet?
8	TA	Det skulle jag göra ja.
9	DS	Vad har du för roll på företaget?
10	TA	Jag skulle säga att jag har nån slags tekniskt ansvarig roll, vi har egentligen inga så superhårt definierade roller kanske, men jag brukar ha en ansvarig roll. Och jag är även partner i bolaget.
11	DS	Arbetar ni agilt?
12	TA	Det skulle jag säga! [skratt]
13	DS	Men ni följer kanske inte manifestet till punkt och pricka?
14	TA	Nej, det gör vi nog inte. men vi har en ganska typisk utvecklingsprocess skulle jag säga, så agilt kanske inte 100% efter boken, men någonting åt det hållet absolut.
15	DS	Hur stort är företaget?
16	TA	Vi är [***] personer tror jag, beroende på hur man räknar.
17	HC	Får jag fråga hur mycket ni omsätter också? Ungefär?
18	TA	Typ [***] förra året.
19	DS	Vad har ni för kunder ungefär?

20	TA	Alltså primärt vill vi egentligen hitta typ startup-bolag som har kommit en bit och kanske har fått lite pengar och har lite kassaflöde, så att de har råd att ta in konsulter helt enkelt, och det är typ mest för att vi tycker att de är roligast att jobba med. Så vi fokuserar på växande bolag som har lätt att göra ändringar i någon mening och vill typ komma snabbt framåt.
21	DS	Så då använder ni NPM-paket i er verksamhet?
22	TA	Ja det gör vi, absolut, det har väl primärt varit till backend-grejer nu, men vi har ju typ kanske några frontend-grejer också, till exempel vår egen sida använder NPM.
23	DS	Så nu går vi över till själva huvuddelen, och den har vi strukturerat lite som ett scenario, eller beskriver olika delar i en process, och hur ni förhåller er till olika moment där liksom. så om vi tänker oss då till exempel att ni har fått ett nytt projekt som ni ska dra igång, hur ser erat riskhanteringsarbete ut i det här inledande skede? Planeringsfasen där. Och då pratar vi om all typ av riskhantering och analys, inte bara NPM-relaterat.
24	TA	Ehm ja [skratt]. Man kan väl säga att vi brukar försöka komma igång så snabbt som möjligt, för att vi har ofta ganska så korta projekt. Och försöker bygga ganska mycket proof-of-concept-grejer eller saker som ska bli en bas för någonting som ska växa senare, och det brukar oftast inte finnas något jättestort intresse från kunden egentligen, att lägga typ mer tid än nödvändigt på någon form av planering skulle jag säga. Så att vi inte brukar ha tid eller råd i en mening att planera jättemycket sånt här i förväg, utan vi får väl egentligen snarare förlita oss på att vi inte gör för dumma grejer liksom. Oftast är de här grejerna kanske inget som sätts i produktion direkt heller.
25	DS	Så man skulle nästan kunna säga att ert arbete är nästan en del av en sorts förberedelsefas för något som kommer senare?
26	TA	Ja, det skulle man typ kunna säga, att antingen så har företagen någon profil eller vad man ska säga eller någon process för hur de gör detta tidigare, till exempel har vi haft en kund där vi typ deployade grejer på deras intranät i någon mening, att de har en skyddad miljö redan som man kommer åt via VPN till exempel. Och då blir det ju direkt, enligt dem, att man får fritt fram. När man ska göra saker som faktiskt går att komma åt från internet och så vidare så blir det en mer komplicerad process kanske, lite mer i framtiden.
27	HC	Ja, det är, förlåt...
28	TA	...ja oftast är det inte en jättestor del av vårt arbete skulle jag säga.
29	HC	Men det är också gällande alla typer av risker där då? Som att gå över tid, eller över budget, eller inte leverera något eller vad det nu skulle kunna va, har ni någon form av tanke kring det då i planeringsfasen alls?
30	TA	Ja, det kan man väl säga, vi försöker typ att så mycket som möjligt jobba på timbasis kanske, för att kunna garantera vår egen säkerhet, så att vi inte får sitta och jobba en massa extra tid obetalt för att kunderna ska bli nöjda. Men det är alltid svårt typ i samma veva som folk inte är beredda att betala så mycket för en säkerhetsanalys är man oftast inte jätteberedd att betala för att lägga flera veckor på, göra en hårt definierad kravspec heller.
31	HC	Nä.

32	TA	Så att så mycket som möjligt försöker vi bara jobba löpande, sen så gäller det kanske att försöka, ändå internt uppskatta hur lång tid saker kommer ta att göra, och ta små steg framåt hela tiden och se till att saker fungerar hela vägen så att man inte riskerar att sitta med något som inte gör någonting när alla pengarna tar slut.
33	DS	Vad känner du Hannes?
34	HC	Jag tror nästan vi får gå vidare, för vi har lite frågor om mer formell riskhantering, men det sker ju inte riktigt i den fasen här kanske.
35	DS	Ni jobbar ju ganska iterativt där, men spikar ni och fattar beslut om arkitektur och val av tredjepartskomponenter som kan bli aktuella att använda i inledningsskedet eller kommer det också under arbetets gång sen?
36	TA	Ja det skulle jag säga att vi gör, men det gäller oftast snarare för oss att välja saker som gör att det går att komma snabbt framåt, och bygga saker så bra och enkelt som möjligt kanske i första hand, och inte kanske så mycket fokus på säkerhet egentligen, framförallt i projekt där det är interna grejer som lyfts.
37	HC	Men har ni riskhantering rent allmänt annars? Alltså identifiering av risk, analys av risk, prioritering och övervakning av olika risker, även utanför NPM och utanför nödvändigtvis tekniska aspekter och säkerhet?
38	TA	Nej, det skulle jag inte säga att vi gör, utan vi är ett ganska litet bolag och har väl inte riktigt kommit dit än.
39	DS	Yes, ska vi hoppa vidare till nästa del i scenariot här?
40	HC	Ja det tycker jag.
41	DS	Så då tänker vi att ni är igång med det här projektet, och att någon utvecklare som arbetar med det här projektet under arbetets gång hittar något paket som skulle kunna vara en möjlig genväg, alltså att man skulle kunna inkludera det som ett beroende istället för att återuppfinna hjulet så att säga, är det fritt fram att göra det eller har ni någon process för granskning innan man inkluderar en sån komponent?
42	TA	Nej vi har ju ingen direkt konkret process skulle jag säga. Det kanske inte är fritt fram heller, vi försöker ju ha så lite externa beroende som möjligt, dels för att det blir lättare att hantera, och dels ur såna här säkerhetsaspekter absolut. Sen beror det lite på kunden också, de har kanske önskemål att man inte ska använda vissa typer av komponenter eller kanske externa tjänster och så där och då respekterar vi det. Men vi förlitar oss lite egentligen på att kunderna definierar vad som är viktigt för dem, sen försöker vi jobba så snabbt som möjligt innanför ramarna för vad dem är ok med. Jag skulle inte säga att det här är något som efterfrågas jättemycket kanske, utan det här är något som, egentligen kanske hade varit vårt ansvar att göra om vi skulle bygga stora publika tjänster så är det kanske ändå vårt ansvar att faktiskt berätta för kunderna om sådana här risker.
43	DS	Men om man tänker lite mer informellt då om man tittar på ett nytt paket, jag misstänker att ni ändå kanske kikar lite på avsändaren och hur populärt det är och så vidare.
44	TA	Ja, försöka minimera antalet paket egentligen och försöka välja stora grejer som många använder, sen skulle jag säga att det finns ju ingen som sitter och tittar på vilka paket de här paketen i sin tur använder, och såna grejer, så nä, det är väl egentligen att gå på någon slags rykte i så fall.

45	DS	Ja, det är lite det vi undersöker också, var i den transitiva kedjan problemet uppstår, eller vem som äger problemet. Men då svarade du nästan på det också, ni har inte heller någon sån här dokumentation utöver över package.json där ni dokumenterar alla komponenter som används?
46	TA	Nej, inte den typen av komponenter i alla fall, det är nog inget vi skulle skriva in i en projektbeskrivning, exakt vilka dependencies allting använder.
47	DS	Vill du ta nästa kanske Hannes?
48	HC	Javisst! Om vi säger att det här systemet nu faktiskt har sjösatts och befinner sig i skarp drift, då undrar vi hur det löpande beroendehanteringsarbetet ser ut? Finns det någon som har en roll som ansvarig för att övervaka uppdateringar och rapporter för eventuella sårbarheter som kan dyka upp?
49	TA	Jag skulle inte säga att vi har någon sån, den typen av underhåll-roll någonstans just nu. Om vi skulle hade det, så skulle jag säga att varje gång man gör någonting just nu, så ser man väl till att försöka uppdatera paket och så i samband med byggen. Som sagt så är formella processer kanske inget vi jobbar jättemycket med än, men ja, det är väl typ så det hade sett ut just nu i alla fall, att man märker eller uppdaterar i samband med annat.
50	HC	Det låter lite som att det är på varje utvecklares egna ansvar att se till att det sköts bra, snarare än att det finns någon som har ett övergripande ansvar gällande de här frågorna?
51	TA	Ja, absolut.
52	HC	Och då undrar jag, hur ser det övriga mer övergripande riskhanteringsarbetet ut då när ett system är sjösatt, alltså då inte nödvändigtvis anknutet till NPM. Finns det någon kommunikation om möjliga risker rent allmänt?
53	TA	Nej, alltså inte allmänt, utan det vi försöker sätta upp är ofta är kanske mer såhär övervakning så att man får reda på ifall systemen går ner, och typ loggning så man kan se vad som har gått fel, och den typen av grejer.
54	HC	Så ni övervakar då möjliga risker i alla fall?
55	TA	Ja, den typen av risker kan man väl säga att vi övervakar, absolut.
56	HC	Det låter lite då som att, det finns ju då en typ av risk, i egentligen all form av utveckling men också då med NPM, är det mer då erfarenhetsberoende, att man är bekant eller har minnen från tidigare projekt, eller man har kompetens med hur man ska arbeta med NPM, är det som väljer hur man ska jobba? Eller det finns inget arkiv eller så där ni samlat på er en lista med risker eller har utvärderingar av vad man ska se upp med?
57	TA	Nej, vi har inte kommit till det än, utan vi går på allas erfarenhet i bolaget med tanke på att vi är ganska små. Men det låter typ som att det vore bra att ha det om man anställer jättemycket folk och framför allt tar in nya utvecklare som inte har så jättemycket erfarenhet.
58	HC	Ja, det är väl det vi lite också undrar, i vilken utsträckning beror storleken på ett företag, eller beror er strategi på hur ni hanterar dessa problemen på vad ni faktiskt bygger och gör, och vilken typ av tjänst ni säljer, men också då hur stora ni är, alltså vilken kapacitet ni har att faktiskt göra de här sakerna?

59	TA	Vi har ganska liten kapacitet att göra de här grejerna. Och vi fokuserar ju typiskt på snabbt växande bolag, av anledningen att de inte bryr sig så mycket egentligen om såna här grejer, och jag tror att i någon mening så ärver vi liksom kundens krav i de här grejerna, så det är väl lite upp till dem att definiera vad som behövs egentligen, så jobbar vi efter det liksom. Man skulle kunna tänka sig att någon hade bett os att göra det, och då hade vi kanske tittat på att faktiskt göra någon process för dem, men ja, annars lever vi på vad de sagt till oss att göra.
60	DS	Vi var inne och rörde vid detta lite innan, det här med uppdateringar in NPM då specifikt, men hur resonerar ni kring det? Om det finns nya uppdateringar, installerar ni dem inte automatiskt men alltid, eller har ni någon filosofi där? För risken är att även uppdateringar kan användas som en attackvektor, och att det slarvas ganska mycket men de här semantiska versionsnumren så att de ibland inte faktiskt är bakåtkompatibla fastän de ska vara det. Är det något ni tänker på eller hur ser du på det?
61	TA	Ja absolut. Alltså det gäller ju att se till att man kan rulla tillbaka uppdateringar främst. Men sen är väl egentligen mycket, det är ju en ganska manuell grej faktiskt att komma ihåg att uppdatera grejer och testa för att se till att det funkar. Och det handlar ju också typ om att ha testet till exempel så att man vet om uppdateringarna tar sönder grejer.
62	DS	Men ni har inget sånt här att vi uppdaterar inte om det inte är en kritisk sårbarhet eller liknande, uppdaterar inte i onödan helt enkelt, utan ni uppdaterar gärna om det finns uppdateringar.
63	TA	Ja det skulle jag typ säga.
64	DS	Då när ni då gör en sån här uppdatering då låter som att ni kollar mest på att den är kompatibel, så att den inte äventyrar någon befintlig funktionalitet då?
65	TA	Ja precis. Det kanske är en grej jag borde snackat om tidigare i riskhantering, att vi försöker skriva typ vältestade grejer, och det hjälper typ med såna här problem att vi kan typ vara någorlunda säkra på att vi inte tar sönder någonting om man uppdaterar grejer till exempel. Så att det skulle jag säga är en stor fördel här, att man kan prova att uppdatera och sen om det skiter sig kan man alltid gå tillbaka, och testa uppdatera nån grej eller undvika någon uppdatering till exempel.
66	DS	Vad känner du Hannes har vi missat någonting?
67	HC	Ja, dels så pratade vi lite om... Det jag glömde fråga dig var, hur tror du det hade sett ut ni var 200 anställda istället? Om jag har förstått det rätt, så är det så att eftersom kunderna inte riktigt ber om den här typen av rigoröst riskhanteringsarbete så har ni egentligen inget behov av att göra det. Men om ni hade varit 200 anställda istället, hade det kunnat se annorlunda ut då?
68	TA	Ja det tror jag. Just nu väljer vi egentligen aktivt bort de här typerna av uppdrag egentligen för det är svårt att hantera. Det kan ju hända att man behöver någon person som typ enbart jobbar med såna här saker. Om vi blev 200 pers så kan man väl tänka sig att vi hade haft någon som var specialist på det här och som man då blandar in lite i varje projekt. Men det är lite svårt att sia om.

69	HC	Ja precis, det är ju spekulativt. Men vad jag undrade är liksom relationen till vad er storlek och vad ni har för kapacitet och vilka kunder ni har. Alltså det är ju lite av en triangel där, hur man arbetar, det beror ju dels på den kompetens ni har, inte kompetens men dels kapacitet, dels vad kunderna har för krav, och då vad som då är bäst att prioritera och arbeta med.
70	TA	Ja. Nä jag skulle säga att det hänger absolut ihop och det kan hända till och med att man ska ta nåt projekt och det kanske sätter typ 50, 100 pers att då blir man nog tvingad i mycket större utsträckning från kunden också, då kommer de också ha större kapacitet från sin sida, det kan finns någon redan på deras företag som tar hand om de här bitarna. Så det problemet har vi inte riktigt när vi jobbar på mindre bolag, om inte de har någon som bryr sig [skratt], eller som jobbar med det så...
71	DS	Det kanske är känsligt, men är den ansvarsfrågan reglerad på något sätt, om de skulle ta ett sånt här system som ni har utvecklat och det skulle sättas i produktion och visa sig innehålla någon sorts katastrofal säkerhetssårbarhet, vem faller ansvaret på i en sådan situation? Vet du det?
72	TA	Vi har ett avtal som reglerar det, där vi typ skyddar oss lite från de här typerna av risker. Jag kan typ inte säga exakt vad det står rakt upp och ned, på rak arm, men vi försöker typ att egentligen skydda oss lite från det.
73	DS	Då kan man ju säga att, för det här med riskbedömningar det blir väl ofta någon sorts, att man väger den eventuella konsekvensen mot risken att den inträffar, men då kan man ju också säga att del så är det kanske inte ni som gör själva bedömningen i de flesta av de här fallen, för det är redan era kunder som har önskemål, men också att det är inte ni som tar konsekvensen kanske så att det blir inte lika...
74	TA	Nä precis. Det skulle man väl absolut kunna säga liksom, det är kanske inte så att vi drar oss undan från ansvaret i någon mening men vi kan liksom inte, alltså för att kunna lämna en sådan garanti till exempel, då hade man behövt göra den här typen av analyser mycket mer ingående, då vi inte kan göra det kan vi inte heller ta på oss ansvaret på det sättet.
75	HC	Ja precis, för det jag också är intresserad av är hur de här olika typer av strategier hänger ihop, alltså strategin för hur man då hanterar NPM, strategin för riskhantering mer generellt då, men också den totala strategin för verksamheten, det låter lite som att det viktigaste är att komma fort framåt ungefär, och då är riskhantering, att ta ett riskhanteringsarbete det gör att det går långsamt, och att använda NPM gör att det går fortare typ.
76	TA	Ja absolut. Och det är lite så vi valt att fokusera på det här, för vi tycker det är roligt att bygga saker snabbt och vi, det är kanske ingen som egentligen vill sitta och planera för mycket grejer och analysera den här typen av grejer, och det vill inte kunderna heller i någon mening, alltså det kan vara någon som vill ha typ en proof-of-concept av sin produkt till exempel, då vill man inte investera i det innan man ens vet om den kommer sättas i produktion över huvud taget. Men det känns om att det är lite hela filosofin i den änden av industrin i alla fall. Eller den här lite mer startupaktiga, bara kör på.
77	DS	Det här med att ni ofta hjälper dem i startfasen av projekt eller av ny produkt, brukar ni då följa med hela vägen till liksom skarp release, eller brukar de lämnas över till någon annan någonstans i den processen? Alltså brukar ni ansvara för drift och underhåll också eller är det mer den här initiala utvecklingen?

78	TA	Det har varit lite 50–50 kanske, det är inget som är i faktisk produktion publikt ännu, utan kanske det som vi byggt som är i produktion är typ interna grejer, och då har vi inte tagit hand om driften för att det finns någon annan som gör det. Men vi har planerat egentligen för att ta hand om drift också. Men det kommer lite längre fram nu.
79	HC	Ja, för att sådana här proof-of-concept-projekt då kan man ju egentligen lämna över lite vad som helst, bara man visar att det funkar att göra det antar jag, huruvida det är säkert eller inte kan jag tänka mig att det inte spelar någon roll?
80	TA	Nej alltså lite så. Vi försöker typ göra någonting som kanske lite mer än bara minsta möjliga utan typ en bas som man skulle kunna bygga vidare på, och ja, vi bygger saker med typ tester och sådana grejer till exempel som man kanske inte hade gjort i bara sån här all out, bara visa liksom, utan tanken är typ att man ska kunna gå från proof of concept vidare till produktion sen, och det hade ju klart varit bra då att sätta någon form av praxis för hur man tar hand om den här typen av grejer också, lite längre fram.
81	HC	Hur länge har ni arbetat med NPM, och hur har det arbetet förändrats över tid?
82	TA	Som bolag har vi egentligen haft och göra med det sedan början, och alla som jobbar här har typ använt det i alla fall ett par år tidigare, så typ, jag kan väl svara mer personligen då, jag vet inte, det har bara blivit en grej, det är verkligen industristandard nu, man kommer ingenstans inom typ web utan att använda NPM, jag vet inte, det är väl också en parameter här att man har typ inte direkt något val annat än att ta på sig dependencies, eller typ bygga allting själv, då blir det ju ofta liksom, om du ställer den frågan till en kund [skratt], så ja ska vi lägga ett par tusen timmar på att bygga liksom React igen, eller ska vi bara installera det på tio sekunder? Det är ingen som kommer betala för de här grejerna. Och det tror jag inte kommer ändras heller, och det är lite så det alltid har varit, att man installerar grejer för att spara tid, och slippa typ göra grejer, och förhoppningsvis få bättre grejer också.
83	HC	Ja, vad hade konsekvenserna blivit om NPM bara försvann? Om man inte hade det? Vad gjorde man innan?
84	TA	Ja, innan var det ju ännu värre skulle jag säga, i någon mening. Man hade ju fortfarande typ dependencies, men då uppdaterade du ju aldrig dem i regel, förutom om det var någonting som de verkligen behövde, eller typ upptäcker att det kommit en ny version manuellt typ av någon anledning, kanske läser i docsen eller ja. men det fanns ju inget smidigt sätt att hålla reda på om det fanns en uppdatering ens, så på det sättet är det väl ändå bättre nu än vad det har varit tidigare kanske.
85	HC	Det är ju det som är rätt spännande, vad litar man mest på, ett paket med en miljon nedladdningar eller 200 rader kod som ens kollega har skrivit, vilket är den bästa koden, vilket är mest säkert?

86	TA	Nej precis, det är den andra parametern. Om du skriver det själv så ja, då kan du ju också göra fel. jag skulle säga att den största risken här är väl egentligen automatiserade attacker typ. Med den här typen av paket, om du skulle hitta någon stor sårbarhet i ett paket som jättemånga använder så kanske du kan göra en automatiserad attack och typ gå på alla som använt den, medans om jag sitter själv och skriver någonting, och inte använder några externa dependencies, kan man säkert hitta typ nåt hål i den, om man letar, men det är antagligen ingen som kommer göra det, beroende på vad det är liksom. Det är just den avvägningen, är det här paketet man installerar, är det ens något som är rimligt att attackera? [Tekniskt fel, ohörbart]
87	DS	Precis, för där gör ni ju någon bedömning också, det kanske handlar om det använd på server eller klient-sidan också?
88	TA	Ja. där är det väl egentligen kanske ett medvetet val att vi brukar använda typ Node/Express, som ändå är typ standard, om det är något fel där, så kommer det påverka en jättestor del av hela internet. Medan om det är någon dependency som används på något ett sätt, inne i, jag vet inte riktigt vad det skulle kunna vara, men bara en liten modul som används någonstans långt inne i din backend, är det ens rimligt att någon skulle kunna komma åt den sårbarheten, det är ju inte alls säkert.
89	DS	Nä, och där kommer man in på det här om det är malicious, alltså om det är med uppsåt, eller om det bara är en lucka som finns av... men ja, det behöver vi inte gräva ned oss i.
90	HC	Det som vi också tycker är rätt spännande, det är hur man avgör hur pass villig man är att ta en risk? Om vi säger att det är låg sannolikhet att det händer någonting, men konsekvensen är väldigt stor, eller hur gör man den bedömningen, för det finns ju ändå vissa risker som företag väljer att utsätta sig själva för, för att det i stort sett är en del av strategin, hur gör man en sån övervägning, finns det risker som då egentligen har högre riktvärden som ändå man är villig att utsätta sig för?
91	TA	Ja det gör det väl absolut, och det skulle kunna vara såna grejer som typ att, välj en bra webbserver till exempel, för det är ändå de grejerna som du kan ansluta till direkt från internet, som det är störst risk att någon faktiskt kan göra någon ordentlig skada. Om du har, du bygger någon typ av microservice som aldrig ens kan bli nådd från det publika internet, då kanske det spelar mycket mindre roll, kanske inte ens bryr dig mycket, för att någon måste ändå ta sig igenom ett annat lager för att ens komma till den. Så jag vet inte, det är ju alltid någon slags undermedveten avvägning i det här även om man kanske inte tänker på explicit på typ pakethantering och sånt där, men säkerhet överlag, om du väljer skydda någonting, då gäller det att välja bra programvara. Och det man brukar välja är saker som används mycket, och helt enkelt förlitar sig på att om alla andra är beredda att ta den här risken så kan jag också göra det.
92	DS	Ja, men jag känner mig ganska nöjd. Vad känner du Hannes?
93	HC	Ja jag också, i alla fall får jag börja med att tacka så mycket för att du ville vara med här.
94	DS	Det ska vi inte glömma!
95	TA	Nej ingen fara
96	DS	Då stoppar jag här nu

Appendix D

Transskript Vårdbolaget.
6/5–2020, via Slack.
Längd: 35 minuter

Deltagare och förkortningar:
Författare: David Strömbäck (**DS**), Hannes Carlsson (**HC**)
Respondent: Front-end-konsult på Vårdbolaget (**FE**)

Rad	Talare	Tal
1	HC	Skulle du kunna beskriva företagets verksamhet?
2	FE	[***] sitter då som ett vårdbolag egentligen. Deras huvudsakliga ämne är att de bedriver (behandling)-kliniker i ett antal länder runtom i världen. Och det är väl egentligen det som är företagets... företagets ide helt enkelt. Så det är ett tjänstebolag kan man väl säga. Där man har sina sjuksköterskor och kliniker, köper in (behandling)-maskiner och tar hand om sina (behandling)-patienter. och då är det ju patienter som haft (åkomma).
3	HC	Men kan man då... Skulle man då kunna säga att, det låter som att man kanske inte kan säga att systemutveckling är en kärnverksamhet?
4	FE	Nej, det är ju inte ett traditionellt it-bolag, det är det inte. Bolaget har sin IT-avdelning men det är inte kärnverksamheten, nej.
5	HC	Vilken roll har du då på [***]?
6	FE	Jag sitter som frontend tech-lead, där jag är tekniskt ansvarig för att utveckla klientapplikationer åt [***] internt.
7	HC	Ah, och hur stort är [***]?
8	FE	Det är en bra fråga...
9	DS	Ca [***] tror jag
10	FE	Om det är totalt antal anställda så har inte jag så bra koll på det faktiskt utan det kan ni nog hitta på [***]s hemsida.
11	HC	Precis, men det ni utvecklar, jobbar med, det är inte för kunder då, utan för internt bruk?
12	FE	Precis, det är för internt bruk
13	HC	Skulle du säga att ni arbetar agilt? Det är en kontrollfråga vi har.
14	FE	Absolut, det gör vi. Våra leveranser sker en gång i månaden och de nya kraven som kommer in kan ju komma med olika mellanrum så det beror ju alltid på behovet. Men vi försöker följa projektmodellens gång så gott det går.
15	HC	Ah, nice. Då slutligen, vår sista kontrollfråga: Använder ni NPM-paket i er utveckling?
16	FE	Ja, det gör vi. Vi använder NPM-paket just nu i samtliga klientapplikationer som vi kör internt.

17	HC	Alright, och då är tanken då följande för majoriteten av den här intervjun: Vi har lite olika scenarion, en processföljd i ett utvecklingsprojekt där vi beskriver lite i olika faser och sen då ställer lite följdfrågor på det. Om man tänker sig att företaget ska dra igång ett nytt utvecklingsprojekt. Hur ser riskhanteringsarbetet ut i det inledande skedet? Vilka aktiviteter finns när det gäller riskhantering? Inte nödvändigtvis anknutet till NPM, men har ni planering, identifiering, analys, prioritering, övervakning av risk?
18	FE	De tyngsta risk- och säkerhetsfrågorna hanteras främst utav DPOn på [***], dvs data Protection officer. Och det är ju också i och med att [***]s interna system hanterar känslig patientdata, men också annan persondata, och där är det ju primärt... den stora frågan är ju GDPR och patientdatasäkerhet, så dessa frågor i en högre nivå hanteras av våran DPO. Sedan så rinner det ner på systemarkitekterna och de ansvariga utvecklarna utav applikationerna, så det är väldigt mycket som präglas av regelverken som finns.
19	HC	Ja, det kan jag tänka mig. Och det är det väl också så, redan i ett planeringsstadie, att man måste ta hänsyn till allt detta?
20	FE	Ja
21	HC	Ja, ni har ju då den här DPOn som då är ansvarig för detta, men kan identifierade risker påverka hur ett projekt planeras? Ni har ju de här sakerna ni måste se upp med, påverkar det egentligen vilken struktur ni har på projektet, vilka val som görs och så? Eller är det isolerat till exakt hur man ska lösa de frågorna?
22	FE	Det påverkar inte så mycket på vilka tekniker det är vi väljer att arbete med, snarare så påverkar det ju mer strukturen och arkitekturen av våra applikationer, dvs hur är det vi hanterar känslig data, och försöker minimera exponeringen utav data så mycket som möjligt. Det är väl snarare det som är den viktigaste punkten i datasäkerheten. Gällande vilka teknikval vi gör: Där skulle jag inte säga att vi låter datasäkerheten spela för stor roll i och med att vi gör ju tekniska val mellan de största och mest standardiserade ramverken och biblioteken som finns där ute. Så gällande säkerheten gör det inte någon större skillnad egentligen om vi skulle använda tekniken ena eller den andra, Utan det viktiga för våran del är hur datat hanteras, och se till att det är ett standardiserat verktyg vi använder.
23	HC	Okej, och då undrar jag. Om man tar det ur ett ännu större perspektiv än persondata och teknisk risk, alltså alla möjliga typer av risker som man kan stöta på, har ni någon form av arkiv eller lista med tidigare risker som man har mött i tidigare projekt? Finns det något register för risker så man vet att "jo, det här ska vi se upp med, så då kollar vi på den här listan och hur man kan hantera dem".
24	FE	Det var en bra fråga, historiskt sett så vet jag att det finns dokumentation på risker och allmänt liksom, riskanalyser som har gjorts i tidigare system som man har byggt. Och det är väldigt mycket av de nyare projekten idag som följer... eller liksom är baserade på det, även om det finns en hel del som har ändrats sedan vår stora plattform byggdes. Men dokumentationen, den äldre dokumentationen vet jag inte var den finns lagrad, men sen så har vi dokument där vi dokumenterar nya riskbedömningar som vi har gjort.
25	HC	Ah, ja, ja precis. Så att det kan uppdateras, det här registerna då?

26	FE	Ja, precis. Jag har lite dålig koll på exakt var vi har dem just nu, och var jag kan komma åt dem, men det ska absolut finnas.
27	HC	Ja
28	FE	Och det är också någonting som vi måste se till att det finns dokumenterat, för det är en väldigt viktig regel i GDPR t.ex., att dokumentation ska finnas för våran riskhantering och vilka beslut vi tagit gällande säkerhet.
29	HC	Ja
30	DS	Yes. Men då skulle man kunna säga att: Då fattas beslut kring arkitektur och val av tredjepartskomponenter ganska mycket i den här inledande planeringsfasen?
31	FE	Arkitekturen framförallt, tredjepartsverktygen absolut, men det är ju också precis som när vi går tillväga i att välja ett ramverk som vi vill bygga våra applikationer i. Där gäller det ju att det är väl beprövade projekt som vi använder oss av. Som är stabila, som använts en längre tid, är inte nödvändigtvis releasat inom den senaste månaden utan det är något som är välbeprövat, vältestat, bevisat att det fungerat och att det inte har några risker. När vi tittar på tredjepartsverktyg är det också väldigt viktigt att det är många som använder och att projektet har levt länge, för det är inte förrän då som vissa säkerhetsrisker kan upptäckas och täppas till.
32	DS	Yes
33	FE	Så återigen, det viktiga är att det är ett större projekt, det är mer eller mindre respekterat som en standard inom utvecklings... inom communityn.
34	DS	Yes, tittar ni också då på avsändaren av ett verktyg eller en komponent, eller tillverkaren/leverantören, vad man nu ska säga.
35	FE	Absolut, det är en väldigt viktig punkt. Vilka är leverantörerna? Och där har vi ju ett set utav leverantörer som vi jobbat med sen innan och vet vad vi känner oss trygga i. Skulle det vara en nyare leverantör som vi inte har jobbat med sedan innan , så är det ju väldigt viktigt att man går in och tittar på antal användare, livslängden på projektet, är projektet fortfarande ett projekt som lever, dvs underhålls det fortfarande. Och då är det ju främst ur risksynpunkt.
36	DS	Ja. Händer det någonsin att ni går, liksom på källkodsnivå, och granskar innehållet i olika paket och komponenter?
37	FE	Nej, det händer inte.
38	DS	Nej. Men vi var inte på det här, vad som väger in. Nu pratade vi om startfasen här, men gäller detta också under ett projekts gång, om en utvecklare t.ex. ser en möjlighet att ta en genväg genom att sig av ett NPM-paket, t.ex.? Är det samma rutin där då, med granskning av de här faktorerna som vi precis pratade om?

39	FE	Jo men precis, vi tar väl främst inte in tredjepartspaket där vi känner oss osäkra med leverantören eller ser att det är ett mindre projekt, utan då känner vi ju främst att.… de beror ju också på komplexiteten såklart, komplexiteten av det verktyget som vi vill använda oss av, är det någonting vi kan replikera eller är det något vi kan lösa själv så gör vi det primärt, om det inte finns någon större leverantör som vi faktiskt kan... som är betrodd så att säga. Men det har ju flera andra anledningar också, drar vi in tredjepartsverktyg så är det inte alltid heller att tredjepartsverktyget bara löser de problemen som vi behöver utan vi vill ju gärna hålla källkoden nere så mycket som möjligt för att minska filstorlekarna som behövs sen, och resurserna som kommer att (ohörbart) över nätverket. Säkerheten är en viktig punkt.
40	DS	Ja, och då antar jag också att: Det är inte en enskild utvecklare som fattar ett sådant beslut, utan det är alltid mer än ett par ögon på det här innan det går i produktion?
41	FE	Precis, som en regel så har vi i [***] att all kod som levereras till produktion ska ha passerat två utvecklares ögon. Då är det ju alltid att det ska finnas minst en som gör kodgranskningar, sen ska det ju finnas testare också såklart. Men det ska alltid vara två personer som kollar igenom koden innan den når produktion.
42	DS	Yes, okej. Om man då tänker, alltså förutom package.json, upprättas det någon dokumentation av olika komponenter som används i de här projekten?
43	FE	Förutom package.json har vi ju package-lock som också versionshanteras, så den går ju lite mer in på djupet och tittar på vilka sub-dependencies har våra primära dependencies som vi förlitar oss på.
44	DS	Ja
45	FE	Förutom det så har vi inte någon större dokumentation, det är väl i så fall om det skulle påverka arkitekturen på något större plan så dokumenterar vi. Våra arkitekturerna beslut har vi sagt att, det vill vi gärna underhålla... eller har en changelog på. Men vi har ingen dokumentation av vilka paket vi använder, bara för spårbarhetens skull, utan där är ju faktiskt package.json och package-lock versionshanterade.
46	DS	Hur säkerställer man fortsatt tillgång, framtida tillgång till de här komponenterna, har ni något lokalt paketregister där sparar versioner som används eller förlitar man sig på det offentliga paketregistret där?
47	FE	Det är både och, det beror på lite vad frågan var exakt, men vi drar in NPM-paket från NPM-registret, det officiella NPM-registret, och så har vi egenpublicerade NPM-paket
48	DS	Frågan gäller egentligen paket från det officiella registret, för det ju varit ett par fall där paket har avpublicerats från det officiella registret, och det har ställt till problem i byggen som förlitar sig på det paketet, och att det därför kan finnas en poäng med att spara ner sina egna lokala kopior av versioner som man använder.
49	FE	Jag förstår, nej det gör vi inte, vi har ju NPM-cacheing i våra ci-setuper, men de lever ju också bara en viss tid. Så vi laddar inte ner binärerna för Node-paketen och sparar någonstans, förutom i cachen.

50	DS	Yes, super. Om vi då går vidare i den här processen och tänker oss att det här systemet befinner sig drift, hur ser det löpande arbetet med hantering av NPM-beroenden ut? Jag tänker på övervakning av nya security advisories och installation av uppdateringar osv. Har ni någon roll som ansvarar för det och hur ser arbetet ut?
51	FE	Nej, det är teamet som ansvarar för den biten. I vissa projekt så har vi automatiserat dependency scanning, där det helt enkelt scannas för varje leverans som vi gör utav paket, det behöver inte nödvändigtvis vara en produktionsleverans, men varje bygge som vi gör så görs det en granskning utav rapporterade risker med de NPM-paketen som vi har installerade. Det är något som vi har planer på att installera i samtliga projekt, just nu så har vi inte det automatiserat överallt.
52	HC	Ja
53	FE	Men ansvaret, det ligget på teamet så att säga. VI har inte en person som är utnämnd att vara ansvarig för den biten. Och sen så kan jag väl också säga att vi har lite olika sätt gällande installation av våra NPM-paket, vissa projekt är lite mer.... Ja vad ska vi säga... flexibla med vilka paket de tillåter installeras, och vissa projekt har fixed versioner där det alltid är samma paketversion som installeras, och sen så lever den vidare till teamet har gjort ett aktivt val att uppdatera dem här paketen.
54	HC	Hur ser det övriga, alltså det mer övergripande, riskhanteringsarbetet ut under projektets gång? Alltså då tänker jag också utanför NPM.
55	FE	Under projektets gång så skulle jag inte säga att det skulle vara mycket mer tanke eller underhåll på säkerheten, förutom då att vi vill automatisera scanningen utav NPM-risker. Vilket område undrar ni om?
56	HC	Egentligen generellt, men främst då, under projektets gång så är det kanske övervakning som är mest aktuellt, och i NPM så sker väl det då genom att man tittar ifall det kommer nya uppdateringar och rapporter, och att då kan status ändras.
57	FE	Ja men precis, så man vill hålla sig uppdaterad med paketen, det är ju nummer ett. Men sen så är det ju också om det skulle upptäckas att det finns en mindre version som har en säkerhetslucka, så kommer ju monitoreringen in och är väldigt viktig där.
58	DS	Om man tänker på risker som, nu kanske inte det är riktigt din roll, men alltså som att projektet skulle bli försenat eller inte kunna leverera av någon anledning? Hur arbetar man med det, ägnar man någon tanke åt sånt också?
59	FE	Om vi nu bara säger risken att ett projekt blir försenat, och sen anledning till vad det blir försenat, det menar ni inte skulle vara en säkerhetslucka?
60	DS	Nej, av någon annan anledning, det kan vara vad som helst egentligen.
61	FE	Nej det gör vi inte just, inte någonting som jag varit delaktig i den närmsta tiden, nej.
62	DS	Nej, alright
63	HC	Ni har inte heller någon... det låter ju som att ni har iallafall implicit prioritering av risk, där det verkar som att det viktigaste är att hantera persondata på ett korrekt sätt?
64	FE	Datasäkerheten är det som är absolut tydligast, ja
65	HC	Ja, men har ni någon annan explicit prioritering av risker, där ni liksom har någon form av lista över "det här är viktigast", eller att man ordnar olika typer av risker på vad det är man främst ska se upp för.

66	FE	Mycket möjligt, men det är inte någonting jag är delaktig i så fall. Nej det skulle jag inte säga, utan som tech-lead och utvecklare så är vi mer delaktiga i datasäkerheten, och självklar då också risker, den tekniska säkerheten som finns. Men någon riskhantering på projektnivå, eller på någon annan nivå, är inte jag delaktig i.
67	DS	Alright. Vi rörde vid det innan, med just uppdateringar till NPM-paket och att ni resonerar lite olika beroende på vilket projekt och sådär. Hur tänker ni just med säkerhetsuppdateringar kontra helt funktionella uppdateringar? Problemet är ju lite det att uppdateringar, även om de ofta är bra av någon anledning, också kan vara en attackvektor, alltså att man får in skadlig kod förklädd som en uppdatering. Hur tänker ni kring det, är ni försiktiga med att uppdatera?
68	FE	Det vi tänker rent generellt är att vi gärna vill att större versionsuppdateringar... vi vill gärna vänta med större versionsuppdateringar innan det har bevisat sig vara säkert, innan det har bevisat sig fungera som det ska. Så att man har en stabil version att arbeta med alltid, det kan man väl säga är det som är gemensamt hos alla projekten. Sen så finns det lite olika skolor gällande om man ska köra löpande versionering precis så som du nämner gällande att man vill ha in den senaste funktionaliteten, jämfört med om man vill köra på en säkrare och tryggare version tills den dagen då man faktiskt gör ett aktivt val att uppdatera sina paket. Där skulle jag säga att det är lite 50/50 i projekten, där vissa gärna sätter att patch och minor-versioner får uppdateras löpande, men det är som du säger, där finns det ju en risk att det kommer in skadlig kod och det är ju också ett ganska så... det är ju inte ett helt ovanligt problem heller, som NPM har haft med, och det är ju en typisk lucka också, som hackare har upptäckt, där man väldigt enkelt kan få ganska så stor spridning av att komprimera (menar sannolikt "compromise") NPM-paket.
69	DS	Precis, det är det som inspirerat oss att försöka undersöka det här lite. Men du sade det innan, att det här med semantiska versionsnummer och löst specificerade beroenden, att det använder ni i vissa projekt medan i andra är det...
70	FE	Ja, I vissa projekt har vi helt fixed versionering och i vissa är det löst.
71	DS	Vad är det som avgör där, är det hur kritiskt det här systemet är, alltså hur pass... T.ex. om det bara används internt, eller om det är uppkopplat mot internet, sådana faktorer?
72	FE	Det som avgör är väl framförallt liksom, hur kritisk är den nya funktionaliteten.
73	DS	Alright

74	FE	De projekten som har fixed versioner, så som vi ser på dem är ju helt enkelt att vi håller oss till de versionerna som vi har bestämt en gång och sen så uppdaterar vi då till en nyare version när vi behöver den nya funktionaliteten, och inte förens dess. Och det finns ju olika aspekter till det, och det är också en underhållsfråga, att se till att vi har löpande uppdateringar minskar ju underhållet i och med att det blir ju mindre ändringar som kommer ofta istället för större ändringar och mer sällan. Det blir ju en annan migrering som man behöver göra tex om man inte uppdaterar sitt paket på ett år, jämfört med om det görs löpande. Oavsett om det kommer "breaking changes" eller inte. Men jag skulle säga att det är mer en fråga om funktionalitet snarare än en fråga om säkerhet i detta fallet. Och varför vissa projekt har valt att köra fixed version, och varför vissa projekt valt att köra rörlig version med fixed major.
75	DS	Alright, för vi pratade ju lite här i början om olika saker som ni tittar på innan ni installerar ett nytt paket, men gör man någon liknande granskning vid en uppdatering? Jag tänker på sådan här saker som du nämnde som att paketet är övergivet, det kan ju hända att paketet har blivit övergivet eller till och med bytt ägare sen sist liksom.
76	FE	Ja... Bytt ägare skulle väl i så fall innebära att paketet blivit flyttat menar du?
77	DS	Nja, det finns vissa mekanismer i NPM för överföring av ägandeskap, det kan handla om att paket är övergivna, men det kan också handla om att man tycker att någon har stulit ens namn och sådär, det är lite komplicerat.
78	FE	Mm, okej. Nej men just om ett projekt har blivit övergivet, absolut, det är ju... det skulle jag säga är en väldigt stark varningsflagga. Och det tror jag inte heller skulle ske heller att vi skulle använda oss av ett sånt paket om vi absolut inte känner att det skulle vara nödvändigt, men där tror jag inte att vi går så långt. Sen så beror det ju alltid på vad man är ute efter exakt, men skulle det vara så att man använder ett övergivet paket så ställer det mycket högre krav helt plötsligt, och då får man väl också göra en djupare granskning så som du nämnde innan, att man skulle kunna gå hur djupt som helst och titta på källkoden.
79	DS	Då är det kanske tom aktuellt är göra sin egen fork för att få kontroll över det hela?
80	FE	Ja precis, och då är ju också frågan, hur stort är paketet? Är det verkligen hållbart? Så gällande uppdateringar av paket så är det ju främst att se till att det fortfarande är ett levande projekt skulle jag väl säga, och sen så är det ju som vanligt, se till att man inte använder det absolut senaste som blev releasat bara någon vecka tillbaka, utan man vill gärna hålla på det lite och se att det utvecklas åt rätt håll.
81	DS	Ja, men då tror jag vi är klara med själv huvuddelen av intervjun. Eller, hur känner du Hannes?

82	HC	Ja, det håller jag med om. Och då har vi lite bredare, eller kanske något återkopplande frågor, och något som jag alltid tyckt var lite intressant är det här med hur pass villig man är att ta en risk. Man kan säga att en risks storlek är ju en relation mellan sannolikheten att något kan utfalla och konsekvensen av ett sådant utfall, så får man någon form av "risk score". Men man kan... det är kanske lite orealistiskt att förhålla sig till risker bara genom ett sånt score utan det finns kanske vissa fall då man är mer villig att ta en risk som har ett högre "score" än en annan, även om den får ett lägre "score". Alltså att man utsätter sig för vissa risker medvetet egentligen. Har du något resonemang kring det och hur ni arbetar med det i er verksamhet? Finns det tillfällen då ni medvetet tar vissa typer av risker men avviker ifrån andra?
83	FE	Nej, vi tar inga medvetna risker från tredjepartsverktyg, utan skulle man ta risker utav tredjepartsverktyg så hade det aldrig kommit produktion, utan då skulle det vara i testsyfte. Att ta alldeles för stora risker hör inte riktigt hemma i den branschen som vi jobbar i.
84	DS	Precis, om man ska kontrastera [***] med andra vi har pratat med så kan man väl säga det att riskerna, om det skulle hända något här är ju ganska... alltså konsekvenserna är ju väldigt stora.
85	FE	Precis, och jag vet inte om jag kan säga det, att vi varit inför ett sådant fall där vi behöver ta en risk, väldigt ofta så kan vi ju hantera det mesta själv. Skulle det vara något större projekt så är det ju också väldigt mycket mindre risker det rör sig om och minimala risker helt enkelt, så nej, vi väljer inte att ta risker med tredjepartsverktyg.
86	DS	Alright
87	HC	Jo, för det tycker jag också är ganska intressant. När vi pratat med olika företag här, och hur man arbetar med NPM, och sen hur deras riskhanteringsarbete ser ut mer generellt, den totala strategin, så tycker jag det oftast går att se en ganska så tydlig röd tråd mellan hur de här olika sakerna hänger ihop. Det låter lite som att det kanske går att se det här också. Ja, ni är väldigt strikta och har tydliga rutiner för hur ni ska arbeta med NPM-paket, och då även för riskhantering, och att det beror på hur verksamheten konkurrerar och den bransch ni är i. Hur ser du på det?
88	FE	Kan du ta om frågan
89	HC	Ja, förlåt, det blev lite långrandigt. [skratt] Jag menar, finns det någon koppling mellan hur ni arbetar med NPM-paket och hur ni arbetar med riskhantering i större mening? Och då även vad strategin är för hela företaget att konkurrera.
90	FE	Så är det ju, vi måste ju alltid tänka på vad det är för applikationer vi jobbar med, och det reflekteras ganska tydligt också i vår riskhantering på teknisk nivå. Så definitivt är det ju så att sättet som vi väljer att jobba med NPM präglas av vilken bransch vi jobbar i.
91	DS	Ja, om vi ska ta några avslutande frågor här om NPM och tidsdimensionen. Hur har arbetet med NPM förändrats över tid? Det är ju ändå ett ganska nytt fenomen.
92	FE	Oh, hur har arbetet med NPM utvecklats över tid?
93	DS	Ja, var du med innan NPM-tiden?

94	FE	Nej, så mycket som det har ändrats egentligen, vi har kört NPM nu så länge vi har kört med... Ska vi se... Nej men några äldre klientapplikationer har jag inte jobbat med, det har jag inte. Det som skiljt sig främst är väl att i början så använde vi oss av NPM-shrinkwrap istället för package-lock.
95	DS	Mm
96	FE	Jag kommer inte på något mer egentligen gällande vad som... nej...
97	DS	Nej
98	FE	Vi har hållit samma bana ett tag nu, och inte några större förändringar, det är väl att vi har privatpaket för våra större komponenter som vi väljer att återanvända. Förutom det så kör vi väl allt annat vi officiella NPM-repot. Hur vi väljer att jobba med våra privata paket är väl kanske något som har ändrat sig lite men det har inte heller legat så länge, så nej, det skulle jag inte säga.
99	DS	Okej, så då bara avslutningsvis, hur ser ni på fördelarna som användning av NPM medför, hur hade arbetet sett ut utan NPM?
100	FE	I så fall att vi skulle använda ett annat pakethanteringsverktyg istället för NPM, eller om du menar att vi skulle helt slopa tredjepartsverktyg via...
101	DS	Ja, man får ju nästan anta om NPM inte fanns så hade det ju funnits någonting annat, men om det inte fanns något pakethanteringsverktyg, i ett hypotetiskt scenario, hade det gått överhuvudtaget att bedriva det arbetet som bedrivs just nu?
102	FE	Absolut men det hade ju inneburit en väldigt stor prestandapåverkan, då hade vi behövt länka in de här tredjepartsverktygen via dokumentet som laddas ner när man öppnar upp hemsidan, så det hade ju påverkat väldigt mycket på prestandan. Sen så är det ju absolut fortfarande möjligt att använda sig av tredjepartsverktyg, det skulle ju definitivt göra en stor skillnad för utvecklarna också.
103	DS	Jag tänker också på det här med uppdateringar och att hålla reda på alla dessa beroenden, administrationen av dem hade kanske blivit knepig?
104	FE	Ja, precis som du säger, om man vill hålla sig uppdaterad så är det väldigt smidigt med de här pakethanteringssystemen och versionshanteringen av det.
105	DS	Yes, jag vad känner du Hannes?
106	HC	Jag är mycket nöjd
107	DS	Jag är också mycket nöjd
108	HC	Ja, tusen tack för att du ville vara med helt enkelt, det uppskattar vi väldigt mycket.
109	DS	Det är väldigt uppskattat

Appendix E

Transskript Säkerhetsbolaget.
11/5–2020, via Google Meets.
Längd: 37 minuter

Deltagare och förkortningar:
Författare: David Strömbäck (**DS**), Hannes Carlsson (**HC**)
Respondenter: Junior Developer (**JD**), CTO och medgrundare (**CTO**) på Säkerhetsbolaget

Rad	Talare	Tal
1	HC	Ja men då kan vi få det inspelat här om du önskar att anonymiseras med en pseudonym, både du som person och företaget?
2	JD	Ja, det kan vi köra på, med tanke på att det är [ohörbart]
3	HC	Ja, exakt. Och då kan vi fråga dig var du har för roll på företaget?
4	JD	Ja, jag jobbar på [Säkerhetsbolaget] och har jobbat där ungefär ett år, jag är juniorutvecklare så jag sitter i backendteamet helt enkelt och skriver kod där. Utvecklar produkten. Jag och min kompis som också började för ett år sen jobbar väldigt mycket med just parsningen av filer och sånt, bland annat då NMP-filer, det är ju en blandning av olika filer vi parsar och analyserar för sårbarheter. Vi sitter mycket med det, och liknande grejer där så det är högst relevant. Ja, och NPM är någonting, vi försöker hela tiden utveckla, så NPM är någonting vi försöker kolla på just nu också och gå vidare till nästa steget liksom så det är högst relevant just det här att ni tar upp det nu, det är kul. Så ja, det är ungefär det jag håller på med, mycket utveckling, och sen samtidigt så pluggar jag i [***].
5	HC	Hur skulle du beskriva företagets verksamhet? Vad är det ni gör?
6	JD	Usch, det är egentligen lättare om [CTO] kommer in sen, men jag kan beskriva lite enkelt så. Alltså hela affärsidén är ju att... Hur mycket vet ni om open source och liknande? Har ni någorlunda koll på...?
7	DS	Det får vi nog ändå påstå att vi har.

8	JD	Ja, jag tänker mig det med tanke på det ni läser. Man kan ju säga så, i open source-världen så är det ju så att när företag utvecklar produkter nu för tiden är det inte så att man utvecklar allting från grunden hela tiden, man vill inte återuppfinna hjulet hela tiden, så man tar in mycket paket, open source-lösningar, för att kunna fixa de problemen som redan är lösta av tidigare personer. Problemet här är att det är svårare att kvalitetssäkra de grejer man hämtar in från andra källor jämfört med det som jag till exempel utvecklar på [***] lokalt liksom, det är lättare att ha koll på vad jag gör och att jag inte använder sårbara funktioner och så vidare. Så det är där problematiken lite ligger, hur ska man se till att de här grejerna man hämtar in är säkra? Så det är där vi kommer in. Affärsidén är att vi ska analysera efter sårbarheter i den koden de hämtar in, i deras dependencies och så vidare, för att se till så att det de hämtar in är säkert så att de kan lita på det, och så att det är lika säkert som allt annat de använder i sina projekt. Just Javascript är ett väldigt stort språk bland våra användare såklart, React, Angular och allting är väldigt populärt. Men vi har massa olika språk, allt från java med Gradle och Maven, Javascript och så vidare.
9		[pling]
10	JD	Och där kom [CTO] också, perfekt.
11	CTO	Oj, sorry!
12	HC	Hej [CTO]!
13	DS	Hej [CTO]!
14	HC	Tack för att du hoppar in här!
15	CTO	Ja, förlåt för att jag är sen.
16	HC	Ja, vad heter det, bara så att du vet har vi en inspelning som pågår, så att du är medveten om det. Vi kan avbryta so fort du vill egentligen.
17	CTO	Ok, ja, men det är bra.
18	HC	Och vi har redan pratat lite smått med [JD] här, och informerat om att vi, ja vi spelar in det här då för transkribering och det är ingen annan som kommer få se den inspelningen, och, vad heter det, ni kommer också anonymiseras helt enkelt, så både era namn och företagets namn kommer vara anonymt i uppsatsen då som den ska användas i.
19	CTO	Ok, mm.
20	HC	Så att ni vet det, och om ni känner att det är något som inte känns bra eller att ni inte vill vara med så kan ni bara höra av er så, innan publicering, så tar vi bort er medverkan ur uppsatsen helt enkelt, det är inga problem. Vi har pratat lite med [JD] precis om företagets verksamhet, han beskrev lite om vad det är ni gör. Ja, men du kan väl fortsätta, kommer inte riktigt ihåg exat var du var någonstans?
21	JD	Ja, jag hade precis, jag hade gett en rätt översiktlig [ohörbart] ungefär med språken och sånt, det är svårt [CTO] kan lägga till lite, men det jag snackar om mest [CTO] var liksom det vi höll på med open source och att vi ska säkra de komponenter som företag hämtar in från andra ställen och så vidare, och vilka språk vi stödjer ungefär, ungefär det vi håller på med har jag förklarat lite snabbt och översiktligt.
22	HC	Då har vi också ett par till snabba frågor, hur stort är företaget ungefär? Hur många anställda är ni?
23	CTO	Nu är vi cirka [***] anställda.
24	HC	Ok, ja. Hur mycket omsätter ni ungefär?

25	CTO	[skratt] Ja, det är bottom low, nej men ja, förra året vad var det runt [***]
26	HC	Skulle ni säga att ni arbetar agilt?
27	CTO	Ja absolut.
28	HC	Ja vi har redan avklarat det men ni använder NPM-paket i er utveckling?
29	CTO	Ja.
30	HC	Ok, nu är tanken så att nu i den här intervjun har vi ett scenario som går i lite olika faser, så förklarar vi lite olika skeden då utav ett projekt i stort sett och så har vi lite frågor om hur ni gör saker i dem...
31	DS	Vi kanske ska säga det till [CTO] också, bara att den situationen vi befinner oss i med vissa tids-begränsningar med den här uppsatsen gör att vi måste nästan behandla [***] som vi har behandlat våra andra respondenter, så vi kommer gå igenom ett liknande scenario som vi gjort med andra företag, snarare än att behandla er som en auktoritet på ämnet.
32	CTO	Ok
33	DS	Yes, det var bara det, förlåt Hannes.
34	HC	Ja det är ingen fara, ja men då kan vi ta första, eller scenariot börjar med att ni egentligen ska dra igång ett nytt utvecklingsprojekt, och då undrar vi hur ser ert riskhanteringsarbete ut i ett så inledande skede, har ni några faser för planering, identifiering av risk, analys och prioritering, övervakning av risker i en planeringsfas?
35	CTO	I samband med open source menar du?
36	HC	Ja, men det kan vara risker som är helt övergripande, det kan vara risk för att man går över budget, eller att man blir försenad eller egentligen vad som helst.
37	CTO	Ja. Nu har vi faktiskt än så länge inte skapat sådär jättemånga nya projekt, utan de projekten vi har börjat på just nu har mest varit att vi haft interns, som har börjat i år på det, och då så får vi en uppfattning om omfattningen och kan avgöra lite vilken nytta de skulle ha om man fullföljde projekten.
38	DS	Så det är lite proof of concept kan man säga?
39	CTO	Ja lite så proof of concept, precis ja.
40	HC	Har ni några tydliga roller eller tilldelning av ansvar inom riskhantering?
41	CTO	Ja, vi har till exempel [medarbetare], som är vår produktansvarige, så han är ju ansvarig för att ta in feedback från potentiella kunder och befintliga kunder, om liksom nya funktioner till exempel, och då avgöra vad som faktiskt efterfrågas. Och det är riskminimering i det fallet att minimera att spendera tid på saker som inte någon vill ha.
42	HC	Ja precis, för det är en stor risk för er, om jag förstår det rätt, att lägga tid på fel saker?
43	CTO	Ja precis, för det är både en tids och kostnadsfråga, det är både och. det gäller att spendera pengar på rätt saker, men även att få ut rätt saker i tid.
44	HC	Har ni någon form av arkiv eller lista med tidigare kända eller identifierade risker som ni använder er av.
45	CTO	Vi använder vår egna tjänst, när det kommer till sårbarheter, och i viss mån licenser också, i den mjukvaran vi hämtar in. I övrigt kör vi inga direkta system. Vi har ju gitlab för att tracka själva utvecklingen, så där kan man ju dra lärdomar också från tidigare projekt.
46	HC	Har ni då i en planeringsfas fortfarande, kan man fatta beslut om arkitektur och val av tredjepartskomponenter i det skedet?

47	CTO	Ja, det brukar vi göra, i det skedet. Då brukar vi, ju, vårt verktyg [ohörbart] kommer ha detta, men i dagsläget går vi mest in manuellt och kollar hur communityn ser ut, till exempel, på en open source komponent, är den är aktivt, är den en stigande eller sjunkande trend i aktivitet på repositoryt till exempel. Och såna saker. Som vi kollar på, då när vi tar in open source-komponenter i nya projekt.
48	HC	Och sen om man då går vidare till att man börjat med själva utvecklingen av det här systemet, eller man sitter då aktivt och utvecklar, om en utvecklare då har en möjlighet att ta en genväg genom att använda ett paket eller ett beroende, får utvecklaren lov att göra det hur som helst eller finns det riktlinjer för det?
49	CTO	Formellt har vi inga riktlinjer [skratt]. Men informellt så brukar vi uppmana just det där att kolla på aktivitet, mycket det här med aktivitet egentligen, se till så att det inte är döende. Det brukar vi nästan diskutera på förhand, tex vi använder ju SCRUM när vi utvecklar, så lite då på sprint-planning, så brukar man faktiskt göra en liten light architecturing av lösningen, och då brukar man ta sånt här i hänsyn. faktisk ganska sällan som vi behöver lyfta in tredjepartskomponenter i detta skedet vi är i just nu. Just nu utvecklar vi väldigt mycket specifik logik till vår tjänst. Vi har till exempel betalningen som är uppkommande att implementera sådant, och det lär väl bli mer tredjepartsmjukvara. Så mycket av det vi har, har vi redan tagit in sedan tidigare i vårt fall, så det är inte så mycket löpande.
50	HC	När man utvärderar de här paketen, är det helt enkelt att man tittar på aktiviteten, men tittar man på andra saker? Tittar man på avsändare också eller?
51	CTO	Inte jättemycket avsändare men typ, mycket aktivitet, och mycket popularitet. Hur många stars, hur många downloads i registryn som det finns.
52	HC	Händer det att man tittar på vad paketen också innehåller så att man utvärderar källkoden?
53	CTO	Nej det gör vi inte.
54	HC	Och det är då den individuella utvecklaren som gör den här utvärderingen?
55	CTO	Ja, precis. En sån sak man brukar kolla på är dokumentation, det är viktigt, så att det faktiskt går att använda. Det sker nästan naturligt, att om det är dold dokumentation så fattar man inte ens hur man ska använda det till att börja med, så då brukar man inte lägga så mycket tid på...
56	DS	Har ni någon form av code-review eller annan granskning innan saker går i produktion så att det är mer än en utvecklare som tittar på det?
57	CTO	Ja, det har vi. Vi brukar ha något som heter feature branches, så man har sin user story, så öppnar man en branch för den, user storyn, även issuen, och sen sitter en eller flera utvecklare och jobbar med denna branch, och sen när de känner sig klara kör vi en review och då brukar vi vara ett antal andra personer som granskar och kollar. Och eventuellt andra stakeholders, det behöver inte bara vara utvecklare, det kan vara designers eller säljare om det har någon sälj-funktion. Och då granskar man bland annat det här med tredjeparts komponenter.
58	HC	Upprättar ni någon egen dokumentation om de här komponenterna också, NPM-paket till exempel? Eller är det bara package.json?

59	CTO	Det är package.json i huvudsak, men sen på vissa dependencies som vårt testramverk, cypress som vi använder oss av, där har vi skrivit egen dokumentation, hur man exekverar tester och sådana grejer, och har även viss dokumentation, nu är detta inte front-end utan back-end, där vi har PHP och composer-paket där har vi dokumentation för databaser, hur man gör migreringar och sådana grejer. Som är just specifika för vårt use case.
60	HC	Hur säkerställer ni en fortsatt tillgång till paket eller beroenden ni inkluderar, har ni något lokalt paketregister.
61	CTO	Nä, det har vi faktiskt inte. Så om alla de registryna gått ner så vore det illa [skratt]. Å andra sidan så ligger det faktiskt klonat lokalt ju, det är så de funkar, man klonar ju ner repona så att, ja.
62	DS	Om allas datorer kraschar samtidigt kanske [skratt], det är ganska långsökt.
63	CTO	Precis, då ligger vi risigt till. Då tror jag nästan vi har större problem.
64	DS	Ja men då använder ni inget såhär, då har ni inte privata paket och så heller som ni använder internt?
65	CTO	Vi har lite privata paket, men då har vi ingen registry, utan då pekar vi bara på repositoryt och så hämtas de där.
66	DS	Men det var väl ungefär det där Hannes?
67	HC	Ja vi kan väl gå vidare lite.
68	DS	Sen känner jag att det kommer lite frågor i slutet jag kommer på lite under intervjuns gång här, så vi kanske inte ska dröja kvar för länge här.
69	HC	Ja då undrar jag, fast det har vi fått svar på allting här tycker jag. Har ni någon form av kommunikation om risker löpande under utvecklingsarbetet eller när systemet är sjösatt?
70	CTO	Det är egentligen mest under code-review. Under tiden utvecklaren jobbar på branchen så har vi inte direkt jättemycket kommunikation, det är mest i början vid designfasen, i sprint planning, och sen vid review när man sett resultatet, sen löpande har vi vår egen tjänst som söker efter sårbarheter, snart open source health också, så vi har något som är på gång. Där vi kollar på active contributors och stars, och så vidare.
71	HC	Ja, så ni använder er egen tjänst för att övervaka riskerna med NPM eller säkerhets...
72	CTO	Precis, precis.
73	HC	Om det då det dyker upp att ett paket ni inkluderat, eller ett beroende ni har, om det kommer någon form av flagga som säger att här finns det ett allvarligt säkerhetshål, vad gör man då?
74	CTO	Då öppnar vi issues på det, automatiskt från verktyget, sen granskar vi det väldigt snabbt manuellt, bedömer riskerna helt enkelt i vårt fall. Sen öppnar vi helt enkelt en branch för denna issuen och ser till att uppdatera dependencien, för oftast finns det redan en uppdatering tillgänglig, åtminstone om den har disclosat på ett vettigt sätt. Eventuellt lösa backward compatability issues som kanske finns. I SCRUM har man då slack time som det heter, som man har möjlighet att jobba på issues som inte finns med i sprinten, så den är typisk sån man bara kan ta in utan varit med i sprinten från början, så man inte behöver vänta på hela sprinten ska vara klar innan man fixar den.
75	HC	Hur hanterar ni uppdateringar generellt, kör ni semantic versioning?

76	CTO	Ja, semantic versioning kör vi mycket. Vi uppdaterar ganska sällan nu för tiden skulle jag säga. Vi uppdaterar i stort sett bara när det faktiskt kommer något problem.
77	DS	Ja, för det här är ju jättespännande, för att, särskilt höra hur ni resonerar kring detta, det har ju varit ett antal incidenter där det har varit skadlig kod förklädd som vissa uppdateringar, är det ett sånt resonemang som ni tillämpar där att man gärna sitter still i båten, eller hur resonerar ni kring det beslutet att gärna vänta lite?
78	CTO	Det är en trevlig bieffekt, att man blir lite skyddad av det. Egentligen beror det på att det finns ofta så mycket backward compatability changes att det blir ett jäkla jobba varenda gång man gör, så då kan man chunka ihop det och göra stor radda.
79	DS	Så ni är inte särskilt oroade för de här säkerhetsaspekterna vid uppdateringar?
80	CTO	Det trackar vi själva, så om versionen vi använder är utdaterad så löser vi det då, annars om det kommer en sårbarhet i en version vi använder, då löser vi det då, men vi behöver inte lösa det om det inte är ett problem. Och det är bara ren tids [ohörbart]
81	DS	Hur ser ni på det där med att vänta med att uppdatera, att man ackumulerar en viss teknisk skuld, har varit ett tema som kommit upp med många respondenter vi har pratat med, att när man då väl bestämmer sig för att uppdatera något man väntat med kan det bli lite jobbigt om man väntat lite för länge kanske.
82	CTO	Jag tror de som ser de problemen, har nog väntat år, tror jag. Men det vet jag många företag som har de problemen. Vi brukar inte vänta år utan…
83	DS	Ok så det är inte de tidshorisonterna vi...
84	CTO	Ja vi brukar uppdatera de flesta dependencies typ en gång i månaden.
85	DS	Ok, då är ni ändå ganska bra på bollen där.
86	CTO	Men förut när vi var mycket mindre då kunde vi uppdatera varje vecka, inga problem, men i takt med att projekt växer så blir det svårare helt enkelt, att vara helt up to date hela tiden.
87	DS	Kan passa på att fråga det också, upplever ni att de funkar bra med semantic versioning i NPM, eller upplever ni att det slarvas lite ibland och att saker som kanske inte är helt bakåtkompatibla ändå släpps som minor och patch till exempel?
88	CTO	Det slarvas en del. [skratt] I NPM-världen. i NPM-världen skulle jag säga att det slarvas en hel del. I PHP-världen är det mycket bättre, det är faktiskt ganska intressant, vet inte varför det är så. Men det är ju också kaos-mycket paket i NPM-världen generellt. Nej men det slarvas en del, men vi försöker ändå använda semantic versioning så mycket som möjligt när vi sätter version constraints och så vidare i vår package.json och så vidare.
89	HC	Om ett paket uppdateras, granskar ni då det paketet på nytt eller litar man bara på att det funkar?
90	CTO	Oftast kollar vi bara så att testerna går igenom faktiskt, och gör lite manuell testning också.
91	DS	Så ni har ganska omfattande automatisk testning då?
92	CTO	Ja.
93	JD	Ja, jättemycket.
94	CTO	Det tar en halvtimme att köra varenda gång.

95	DS	[skratt] Oj!
96	CTO	Men det sparar tid ändå. Man behöver ju inte sitta och glo på den i en halvtimme.
97	DS	Har ni det integrerat i någon CI-pipeline också?
98	CTO	Ja precis, så vi använder gitlab, och där finns inbyggd CI där som vi använder, den exekverar då [ohörbart], där vi kör då tester i.
99	DS	Förlåt, jag borde kanske ha pluggat på det, men är eran tjänst också möjlig att inkludera i CI-pipelinen så man får skanningen där?
100	CTO	Ja precis, en av våra pipelines är vår egen tjänst.
101	DS	Jag tänkte också fråga, för vi var inne på det innan att ni jobbar med en massa andra beroendehanteringssystem också, hur ser ni på skillnaderna på NPM och de andra? Särskiljer sig NPM på något sätt?
102	CTO	Vi använder i huvudsak två andra. Vi har python där vi använder pip...
103	DS	Nu kanske det hackar.
104	CTO	Hör du mig?
105	DS	Ja nu!
106	JD	Ja vi hör dig nu.
107	CTO	Jag vet inte vad som hände, alla mina skärmar blev svarta. Alltså vi har python, där har vi pip, jag har själv inte använt det så mycket så jag kan inte säga någonting, men rent generellt så funkar det mycket sämre för jag tror inte ens man kan ha version-ranges och sånt.
108	DS	Eller jag tänkte mycket på det här med graden av sammankoppling och beroendedjup och så vidare i de olika systemen. det är väl en kulturell fråga om hur benägna utvecklare är att använda så kallade triviala byggstenar.
109	CTO	Jag undrar nästan om det krävs mer i Javascript än till exempel PHP, att språket i sig kräver att man har, att så mycket elementära funktioner inte finns i språket.
110	DS	Det har vi med, det är ganska uttalat av Brendan Eich där att han vill ha att standardbiblioteket förblir så minimalt som möjligt.
111	CTO	Ok, det förklarar nog den... För i PHP, där med composer, vi ser ju att djupet är mycket mindre, det är ett mycket mindre dependency djup generellt sett, och det underlättar väldigt mycket när det är så.
112	HC	Jag har en fråga som är ganska annorlunda, om man byter ämne, och det är i stort sett vad ni har för typ av kunder, och hur ni arbetar för att kunna konkurrera och få kunderna, vad är strategin då för att kunna leverera värde?
113	CTO	Ja, framförallt har vi kunder som är en viss storlek. Det krävs att kunderna är minimum 20 utvecklare på företagen, innan de börjar känna problem med lite av de här frågeställningarna vi har här. Just det här att hålla dependencies uppdaterade och så vidare, och faktiskt bry sig om security issues, sen finns det vissa branscher som bryr sig mer om detta än andra, såsom med-tech, som är en jättebra sektor som bryr sig om detta, och fintech också. Så det är lit den typen av företag där det är känsligt att data skulle läcka ut. Och just då att man har inte tid eller ork att ta tag i detta om man är färre än 20 anställda. Det har vi märkt tydligt på sälj. Vad var nu den andra frågan?

114	HC	Ja, hur då ni arbetar för att kunna serva de här kunderna, finns det några särskilda omständigheter då, tycker att vi tidigare har märkt att många har, eller att vissa företag har i stort sett ett krav på sig att det får inte ta för lång tid, man kan inte lägga tid och pengar på stor planering eller göra något riskhanteringsarbete, hur ser det ut hos er?
115	CTO	Det är lite, det är därför vi försöker automatisera det så mycket som möjligt egentligen. Vi har inte heller tid och energi att lägga på detta. Om det skulle behöva göras manuellt liksom. Så det gäller att automatisera så mycket som möjligt, det är även vad kunderna efterfrågar, det finns inte en chans att... Om det är riktigt stora kunder, typ över 1000 anställda, då sker en hel del manuellt arbete, eller oftast har de säkerhets experter in house, men innan dess har man inte tid med det men man har ändå behovet, av att faktiskt kolla på dessa sakerna. Så det faller lite mellan stolarna om man inte har ett verktyg som sköter jobbet åt oss.
116	HC	Precis, och då använder ni i er egen verksamhet ert eget verktyg just för att kunna hjälpa er med det arbetet?
117	CTO	Precis, för nu närmar vi oss också det här gränslandet, mellan 20 och 1000 [skratt].
118	HC	Sen undrar jag också, om man tittar på en tidsdimension hur arbetet med NPM har förändrats över tid, nu vet jag inte riktigt hur länge ni har funnits.
119	CTO	Vi har funnits sedan man 2018.
120	HC	Ja precis, jag hade för mig att det var då någon gång. Har ni några processer som har förändrats över den tiden?
121	CTO	Dels har vårt verktyg blivit bättre. Men vi har även, det är lite som jag sa att vi körde uppdateringar mycket mer frekvent tidigare, än vad vi gör idag, och det är bara för att det är svårt att få tiden hålla sig helt uppdaterad. Så det är väl mest det som har förändrats. vi har mer, vi har högre test coverage nu än tidigare, så då kan vi testa lite mindre manuellt nu och ändå känna oss bekväma med att det kommer att funka, det var något man fick testa mer manuellt tidigare.
122	HC	Vi var lite inne på det, men vad är den största risken för er verksamhet egentligen vid utveckling av det här verktyget?
123	CTO	Största risken?
124	HC	Ja, över huvud taget, det kan vara precis vad som helst.
125	CTO	Ja det är en bra fråga, det är en stor fråga. Största risken är väl bara att se till att få in tillräckligt många kunder, det är väl den största risken liksom.
126	DS	Att ingen skulle vilja använda är kanske den största?
127	CTO	[skratt] Ja, det är väl den största risken i vårt skede, skulle jag säga.
128	DS	Du var inne på det innan, att ni pratar med massa kunder såklart, som kommer till er, och kanske inte alltid är jättekunniga inom de här frågorna själva, upplever ni en ökad medvetenhet kring den här problematiken den senaste tiden?
129	CTO	Ja, det verkar som företag blir mer uppmärksammade på detta de senaste åren. Många av oss var en del av ett forskningsprojekt innan på [***], som [JD] kanske nämnde, och bara under de åren, det startade 2015, bara under de tre åren, då märkte vi att det hände väldigt mycket, alltså företag fick upp ögonen för detta.

130	DS	Men ni upplever fortfarande att det är större fokus från vissa branscher, du nämnde med-tech och fin-tech, medan andra kanske inte tar det på lika stort allvar?
131	CTO	Nej precis, så är det absolut. Det smärtar inte i de andra fallen.
132	DS	Det där är ju jättespännande, tror ni det har och göra med dels lagstiftning som de har att förhålla sig till och sen krav på dokumentation, eller vad kan det vara för faktorer som gör att de måste ta det på så stort allvar.
133	CTO	Det har ju mycket med lagstiftning att göra, framför allt när det kommer till med-tech, det finns ju en lag i USA, eller framför allt Kalifornien vilket innebär USA, ni känner kanske till NVD?
134	DS	Ja, just det.
135	CTO	Som är en stor sårbarhetsdatabas, som man måste, för att få lov att sälja en medicinsk enhet i USA så måste man veta vilka sårbarheter man har enligt NVD. Så om det finns någon sårbarhet i NVD som du har, så måste du veta detta, så sådana krav finns det till exempel. Och det är väldigt gynnsamt för oss. Och det är på gång liknande krav i EU också.
136	DS	Alright. Ja, vad känner du Hannes?
137	HC	Ja, det var någonting... Vi pratade lite om att ni hade sprint-planning, kan du utveckla lite mer om vad ni gör i den planeringsfasen i sprinten?
138	CTO	Ja, [JD] du har suttit tyst nu ett tag, du kan väl ta denna, du som är med, jag är inte med i SCRUM-teamet längre.
139	JD	Ja precis, såsprint-planeringen, innan sprint-planeringen så har ju vår SCRUM-master och produktägaren [***] då har de gått in och kollat på tjänsten vad kunder begär och sånt och skapat olika user stories och olika grejer som känns relevanta att vi ska fixa i framtiden och sen prioriterat dem i en slags kö för att se att det här är mest relevant, det måste vi fixa nu, och det här är lite mindre relevant och så vidare. Så när vi väl kommer in till sprint planeringen då har vi redan det fixat. I sprint planeringen så går vi genom det som ett lag, från toppen och nedåt, man diskuterar varje user story och issue och kollar, vad ska göras, hur ser det ut, sen så bryter vi själva ner det hela laget, i olika tasks, och det är då här [CTO] menar att man kollar igenom vad krävs och fixa issues, finns det några problem och så vidare, och det är hela lösningsgrejen där. Ja sen bryter vi ner det där, kollar vad som är rimligt, estimerar hur lång tid eller hur mycket effort det tar. Ja sen väljer man då, hur mycket vill vi göra den här sprinten. Sen väljer man hur många man ska göra och så vidare. Och det är ungefär hela processen, som det går till i den planeringsfasen.
140	HC	Ja exakt, för som jag har förstått det så är det att man i agil utveckling, i SCRUM ofta inte har explicit uttryckt eller formaliserat riskhanteringsarbete, men att det ofta finns där implicit, och att då i sprintar att man har olika möten, att man kanske inte säger att nu ska vi prata om risker, men att man kanske ändå har något som kanske liknar brainstorming om risk eller att man ändå kommunicerar om det kanske. Det känns som att om man prioriterar tasks att det kan man lika gärna göra med risker då kanske, händer det?

141	JD	Det skulle jag nog vilja säga, det känns lite som att hela riskhanteringen, hela processen, det är något som vävs in i hela processen, ända från att [***] och SCRUM-mastern planerar in vad vi ska göra till att vi sitter där och, det blir aldrig explicit att man diskuterar just som du säger, men det kommer hela tiden in i diskussionen. Jag skulle helt klart säga som du säger där att det är något som vävs in på något sätt ändå.
142	DS	En grej som jag kom att tänka på, nu blir det tvärt kast här igen, men som också kommit upp mycket i våra intervjuer, vi var inne på det lite innan, det här med open source och, hur ser ni på det här med säkerheten i att utveckla själv vs att använda lite mer etablerade paket, det blir lite security by obscurity vs granskningen som ändå sker inom open source?
143	CTO	Security by obscurity tror vi inte alls på.
144	DS	Det tror ni inte alls på?
145	CTO	Nej. Vi har ju [***], han jobbar hos oss bland annat, som säkerhetsexpert, så jag hade aldrig kunnat säga att vi jobbar enligt security by obscurity. Men vi tror absolut på att det är bättre med granskningen i open source communityn. Vi litar inte på att vi själva skriver säker kod.
146	DS	Nu blev jag lite avbruten här, nu är jag tillbaka. Nu känner jag mig ganska nöjd tycker jag.
147	HC	Ja jag känner mig också nöjd. Kanske vi bara ska tacka för oss kanske? Tusen tack för att ni ville vara med.
148	DS	Ja det kanske vi ska göra, tack så jättemycket!
149	CTO	Tack själva! Bra frågor.
150	HC	Ja, och som sagt, om ni har några synpunkter, eller inte vill vara med eller något sånt så kan ni bara höra av er.
151	CTO	Absolut.
152	HC	Tack igen, ni får ha det bra!
153	JD	Detsamma!
154	CTO	Ha det bra!

Referenser

Abdalkareem, R., Nourry, O., Wehaibi, S., Mujahid, S. & Shihab, E. (2017). Why Do Developers Use Trivial Packages? An Empirical Case Study on NPM, in *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering - ESEC/FSE 2017*, The 2017 11th Joint Meeting, Paderborn, Germany, 2017, Paderborn, Germany: ACM Press, pp.385–395, Available Online: http://dl.acm.org/citation.cfm?doid=3106237.3106267 [Accessed 3 April 2020].

Agrawal, R., Singh, D. & Sharma, A. (2016). Prioritizing and Optimizing Risk Factors in Agile Software Development, *2016 Ninth International Conference on Contemporary Computing (IC3), Contemporary Computing (IC3), 2016 Ninth International Conference on*, pp.1–7.

Boehm, B. (1988). A Spiral Model of Software Development and Enhancement, *Computer*, vol. 21, no. 5, pp.61–72.

Boehm, B. (2010). A Risk-Driven Decision Table for Software Process Selection, in J. Münch, Y. Yang, & W. Schäfer (eds), *New Modeling Concepts for Today's Software Processes*, Berlin, Heidelberg, 2010, Berlin, Heidelberg: Springer, pp.1–1.

Boehm, B. & Bhuta, J. (2008). Balancing Opportunities and Risks in Component- Based Software Development, *IEEE Software*, vol. 25, no. 6, pp.56–63.

Boehm, B. & Turner, R. (2003). Using Risk to Balance Agile and Plan-Driven Methods, *Computer*, vol. 36, no. 6, pp.57–66.

Bryman, A. (2012). Social Research Methods, 4. ed., Oxford: Oxford University Press.

Chen, C., Lin, S., Shoga, M., Wang, Q. & Boehm, B. (2018). How Do Defects Hurt Qualities? An Empirical Study on Characterizing a Software Maintainability Ontology in Open Source Software, *2018 IEEE International Conference on Software Quality, Reliability and Security (QRS), Software Quality, Reliability and Security (QRS), 2018 IEEE International Conference on, QRS*, pp.226–237.

COSO. (2004). COSO Enterprise Risk Management — Integrated Framework, Available Online: https://www.coso.org/Pages/erm-integratedframework.aspx [Accessed 2 April 2020].

DeBoer, E. (2020). What Is a Package Dependency Manager?, *Sonatype Blog*, Available Online: https://blog.sonatype.com/what-is-a-package-dependency-manager [Accessed 5 November 2020].

Decan, A., Mens, T. & Claes, M. (2017). An Empirical Comparison of Dependency Issues in OSS Packaging Ecosystems, in *2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER), Klagenfurt, Austria, February 2017, Klagenfurt, Austria: IEEE, pp.2–12, Available Online: http://ieeexplore.ieee.org/document/7884604/ [Accessed 21 April 2020].

Decan, A., Mens, T. & Constantinou, E. (2018a). On the Impact of Security Vulnerabilities in the NPM Package Dependency Network, in *Proceedings of the 15th International Conference on Mining Software Repositories - MSR '18*, The 15th International Conference, Gothenburg, Sweden, 2018, Gothenburg, Sweden: ACM Press, pp.181–191, Available Online: http://dl.acm.org/citation.cfm?doid=3196398.3196401 [Accessed 10 March 2020].

Decan, A., Mens, T. & Constantinou, E. (2018b). On the Evolution of Technical Lag in the NPM Package Dependency Network, in *2018 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, 2018 IEEE International Conference on Software Maintenance and Evolution (ICSME), Madrid, September 2018, Madrid:

IEEE, pp.404–414, Available Online: https://ieeexplore.ieee.org/document/8530047/ [Accessed 25 March 2020].

DeGroat, T. J. (2019). The History of JavaScript: Everything You Need to Know, *Springboard Blog*, Available Online: https://www.springboard.com/blog/history-of-javascript/ [Accessed 5 November 2020].

Duan, R., Alrawi, O., Kasturi, R. P., Elder, R., Saltaformaggio, B. & Lee, W. (2020). Measuring and Preventing Supply Chain Attacks on Package Managers, *arXiv:2002.01139 [cs]*, [e-journal], Available Online: http://arxiv.org/abs/2002.01139 [Accessed 2 April 2020].

ESLint. (2018). Postmortem for Malicious Packages Published on July 12th, 2018, Available Online: https://eslint.org/blog/2018/07/postmortem-for-malicious-package-publishes [Accessed 31 March 2020].

Friedman, N. (2020). NPM Is Joining GitHub, *The GitHub Blog*, Available Online: https://github.blog/2020-03-16-NPM-is-joining-github/ [Accessed 17 March 2020].

GitHub. (2019). State of the Octoverse, Available Online: https://octoverse.github.com.

GitHub. (n.d.). About Security Alerts for Vulnerable Dependencies, Available Online: https://help.github.com/en/github/managing-security-vulnerabilities/about-security-alerts-for-vulnerable-dependencies [Accessed 15 May 2020].

Grander, D. & Tal, L. (2018). A Post-Mortem of the Malicious Event-Stream Backdoor, *Snyk Blog*, Available Online: https://snyk.io/blog/a-post-mortem-of-the-malicious-event-stream-backdoor/ [Accessed 15 May 2020].

Hammad, M. & Inayat, I. (2018). Integrating Risk Management in Scrum Framework, *2018 International Conference on Frontiers of Information Technology (FIT), Frontiers of Information Technology (FIT), 2018 International Conference on, FIT*, pp.158–163.

Hammad, M., Inayat, I. & Zahid, M. (2019). Risk Management in Agile Software Development: A Survey, *2019 International Conference on Frontiers of Information Technology (FIT), Frontiers of Information Technology (FIT), 2019 International Conference on*, pp.162–1624.

Hauge, Ø., Cruzes, D. S., Conradi, R., Velle, K. S. & Skarpenes, T. A. (2010). Risks and Risk Mitigation in Open Source Software Adoption: Bridging the Gap between Literature and Practice, *Open Source Software: New Horizons*, p.105.

Hoda, R., Salleh, N. & Grundy, J. (2018). The Rise and Evolution of Agile Software Development, *IEEE Software, Software, IEEE, IEEE Softw.*, vol. 35, no. 5, pp.58–63.

Hussain, T. (2017). Problems in Current Approaches for Risk Identification and Risk Analysis, *2017 International Conference on Computational Science and Computational Intelligence (CSCI), Computational Science and Computational Intelligence (CSCI), 2017 International Conference on, CSCI*, pp.999–1003.

Islam, S., Mouratidis, H. & Weippl, E. R. (2014). An Empirical Study on the Implementation and Evaluation of a Goal-Driven Software Development Risk Management Model, *Information and Software Technology*, vol. 56, no. 2, pp.117–133.

ISO, IEC & IEEE. (2004). 16085 Standard for Software Engineering - Software Life Cycle Processes - Risk Management, *IEEE/IET Electronic Library (IEL); 16085 Standard for Software Engineering - Software Life Cycle Processes - Risk Management*, IEEE / Institute of Electrical and Electronics Engineers Incorporated.

Medhioub, M., Kim, T.-H. & Hamdi, M. (2017). Adaptive Risk Treatment for Cloud Computing Based on Markovian Game, *2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC), Consumer Communications & Networking Conference (CCNC), 2017 14th IEEE Annual*, pp.236–241.

Microsoft. (2018). Microsoft Completes GitHub Acquisition, *Official Microsoft Blog*, Available Online: https://blogs.microsoft.com/blog/2018/10/26/microsoft-completes-github-acquisition/ [Accessed 15 May 2020].

Mishra, B.K., Rolland, E., Satpathy, A. & Moore, M. (2019). A Framework for Enterprise Risk Identification and Management: The Resource-Based View, *Managerial Auditing Journal*, vol. 34, no. 2, pp.162–188.

Npm inc. (2016). Package Install Scripts Vulnerability, *The NPM Blog*, Available Online: https://blog.NPMjs.org/post/141702881055/package-install-scripts-vulnerability [Accessed 15 May 2020].

Npm inc. (2017). `crossenv` Malware on the NPM Registry, *The NPM Blog*, Available Online: https://blog.NPMjs.org/post/163723642530/crossenv-malware-on-the-NPM-registry [Accessed 15 May 2020].

Npm inc. (2020a). About NPM, Available Online: https://www.NPMjs.com/about [Accessed 17 March 2020].

Npm inc. (2020b). Auditing Package Dependencies for Security Vulnerabilities, Available Online: https://docs.NPMjs.com/auditing-package-dependencies-for-security-vulnerabilities [Accessed 7 April 2020].

Npm inc. (2020c). NPM-Update, Available Online: https://docs.NPMjs.com/cli-commands/update.html [Accessed 15 May 2020].

Npm inc. (2020d). Reporting a Vulnerability in an NPM Package, Available Online: https://docs.NPMjs.com/reporting-a-vulnerability-in-an-NPM-package [Accessed 4 July 2020].

Oates, B. J. (2006). Researching Information Systems and Computing, London: SAGE.

Prasad, H. M., Vinod, G. & Sanyasi, R. V. (2015). Risk Management of NPPs Using Risk Monitors, *International Journal of Systems Assurance Engineering & Management*, vol. 6, no. 2, p.191.

Sherer, S. A. (1995). The Three Dimensions of Software Risk: Technical, Organizational, and Environmental, *Proceedings of the Twenty-Eighth Annual Hawaii International Conference on System Sciences, System Sciences, 1995. Proceedings of the Twenty-Eighth Hawaii International Conference on*, vol. 3, p.369.

Siddique, L. & Hussein, B. A. (2014). Practical Insight about Risk Management Process in Agile Software Projects in Norway, *2014 IEEE International Technology Management Conference, Technology Management Conference (ITMC), 2014 IEEE International*, pp.1–4.

Snyk. (2017). The State of Open Source Security, Snyk, Available Online: https://snyk.io/wp-content/uploads/The-State-of-Open-Source-2017.pdf [Accessed 15 May 2020].

Söderland, J., Geraldi, J., Nilsson, A. & Wilson, T. (2012). Reflections on Barry W. Boehm's "A Spiral Model of Software Development and Enhancement", *International Journal of Managing Projects in Business*, vol. 5, no. 4, pp.737–756.

Tang, Y., Zhou, D. & Chan, F. T. S. (2018). AMWRPN: Ambiguity Measure Weighted Risk Priority Number Model for Failure Mode and Effects Analysis, *IEEE Access, Access, IEEE*, vol. 6, pp.27103–27110.

Vaidya, R. K., De Carli, L., Davidson, D. & Rastogi, V. (2019). Security Issues in Language-Based Sofware Ecosystems, *arXiv:1903.02613 [cs]*, [e-journal], Available Online: http://arxiv.org/abs/1903.02613 [Accessed 25 March 2020].

Williams, C. (2016). How One Developer Just Broke Node, Babel and Thousands of Projects in 11 Lines of JavaScript, *The Register*, 23 March, Available Online: https://www.theregister.co.uk/2016/03/23/NPM_left_pad_chaos/ [Accessed 15 May 2020].

Yarn. (2020). Configuration Options, *Yarn - Package Manager*, Available Online: https://yarnpkg.com/configuration/yarnrc [Accessed 5 November 2020].

Yegulalp, S. (2016). Brendan Eich: JavaScript Standard Library Will Stay Small, *InfoWorld*, Available Online: https://www.infoworld.com/article/3048833/brendan-eich-javascript-standard-library-will-stay-small.html.

Zimmermann, M., Staicu, C.-A., Tenny, C. & Pradel, M. (2019). Small World with High Risks: A Study of Security Threats in the NPM Ecosystem, *arXiv:1902.09217 [cs]*, [e-journal], Available Online: http://arxiv.org/abs/1902.09217 [Accessed 21 April 2020].