Myndigheten för samhällsskydd och beredskap

Avtalsbaserad hantering av gemensamma begränsade resurser i samband med kris

FORSKNING

Diarienummer: 2009-4137

MSB:s kontaktpersoner:
Ebba Hallsenius, 010-240 42 33
ebba.hallsenius@msb.se

Innehållsförteckning

Sammanfattning

I ett förmågebaserat krishanteringssystem där flera aktörer oberoende av varandra formar planer för resursaktivering i samband med kris, kan konflikter i resursutnyttjande uppstå. Det kan ha förödande konsekvenser för den gemensamma krishanteringsförmågan. Projektet har tagit fram en kalkyl och en prototypimplementering för att möjliggöra upptäckt av potentiella resurskonflikter mellan flera aktörer genom analysera aktörernas resursaktiveringsplaner. Att hitta sådana potentiella konflikter är beräkningsmässigt komplext. Projektet har fört samman kunskap både från krishanteringsdomänen och från formella metoder, för att kunna ta fram en effektiv algoritm baserad på simulering för att lösa problemet. Resultatet i projektet riktar sig till aktörer med övergripande ansvar för planering för resursaktivering i samband med kris. Den stora potentialen med resultatet ligger i att tillämpa kalkylen på en specifik domän och därigenom kunna göra domänanpassade antaganden som reducerar komplexiteten på det generella problemet.

1. Introduktion

Projektet undersöker en viktig aspekt av förmågebaserad krishantering: nämligen den att kunna avgöra om en förmågebaserad sammansättning av flera system leder till en sammantagen förmåga som kan tillgodose ett uppkommet insatsbehov. Specifikt fokuserar projektet på upptäckande av möjliga konflikter kring resursutnyttjande vilka kan uppstå då flera behov behöver tillgodoses samtidigt.

Det svenska krishanteringssystemet utvecklas mot att bli alltmer tjänste- och förmågeorienterat. I ett förmågebaserat system ägs eller kontrolleras de resurser som behövs vid insatser nödvändigtvis inte av de som är ansvariga. Snarare sätts insatsgrupper samman av komponenter från flera håll då ett behov uppstår. Sammansättningen bestäms då utifrån karaktären på den kris som ger upphov till behovet. De erforderliga resurserna kan t.ex. ägas av privata aktörer, av andra myndigheter än den som behöver den, eller till och med gemensamt av flera aktörer. Efter avslutad insats avvecklas insatsgruppen och resurserna återställs till respektive ägare. Vid ett uppkommet insatsbehov styrs resursaktiveringen dels av resurstillgänglighet, dels av de mekanismer som finns för att koppla en resurs till ett visst behov. En sådan mekanism är så kallad *lös koppling* mellan system där en av de bärande idéerna är att kopplingen av en resurs till ett behov inte behöver vara förutbestämd. Snarare ska man kunna förlita sig på att lämplig resurs ska kunna aktiveras baserat på en matchning mot behovet. På så vis blir aktiveringen inte beroende av en specifik resursinstans. En annan mekanism för koppling mellan behov och resurs är genom *avtal* mellan den som har behovet och den som äger resursen. Avtal kan ingås t.ex. mellan myndigheter, mellan privata aktörer, eller mellan myndigheter och privata aktörer. Exempel där sådana avtalsbaserade kopplingar kan återfinnas är vid entreprenad av snöröjning och vid transporter vid evakuering av medborgare från oroshärdar. Både lös koppling och avtal ger möjlighet till decentraliserad resursaktivering, vilket är en grundläggande princip i förmågebaserade operationer.

För att koppla en resurs till ett behov krävs att kopplingsmekanismerna – lös koppling, avtal, eller någon annan mekanism – är korrekt formulerade eller konfigurerade. I fallet med lös koppling gäller att det behövande systemet har rätt till att utnyttja resursen. I fallet med avtal gäller att avtalen är korrekt skrivna och att de efterlevs. I båda fallen fallerar resursaktiveringen om resursen inte är tillgänglig. Otillgänglighet kan bero på många saker: t.ex. att alla kopplingsbara resurser redan är upptagna; att de är trasiga; eller att resursen måste konfigureras (exempelvis transporteras till en given plats) och att det skulle ta för lång tid för att behovet ska kunna tillgodoses i tid.

Många av de resurser som används i krishantering är begränsade i flera avseenden. De kan ha en viss räckvidd; de kan vara begränsade i antal eller volym; det kan vara ett begränsat antal operatörer som får eller kan använda en viss resurs; det kan finnas begränsning på hur många användare som kan

använda en resurs samtidigt; det kan finnas fysiska begränsningar såsom uthållighet eller att resursernas effekt degraderas med tiden; eller det kan föreligga omständigheter som gör att en resurs inte kan eller får användas vid ett visst tillfälle. Vid sammansättning av en förmågebaserad insatsgrupp spelar sådana begränsningar en avgörande roll i gruppens sammantagna förmåga.

Insatsbehov uppkommer under flera olika omständigheter. Vissa insatser kan det planeras för (t.ex. övervakning av idrottsevenemang), medan andra insatsbehov uppstår mer oförutsägbart. Man kan t.ex. inte veta i förväg när trafikolyckor eller hot mot skyddsvärda objekt ska inträffa. Man kan genom erfarenhet och modeller dimensionera samhällets beredskap för händelser som kräver insatser och de befintliga resurserna är dimensionerade för att kunna tillgodose ett stort antal potentiella behov. Det är dock lika svårt att i förväg kunna säga exakt vilka behov som kan komma att uppstå, som det är att säga när behoven ska uppstå.

Utöver det att man inte i förväg kan säga när och exakt vilka behov som kan uppstå, finns i ett system med decentraliserad resursaktivering ytterligare ett problem. Det är att de befintliga resursaktiveringsplanerna kan stå i konflikt med varandra. Om de som planerar för resursaktivering gör det oberoende av varandra, så finns det ingen garanti för att två skilda aktörer inte har inkluderat samma resurs i sina planer. Det behöver i sig inte utgöra ett problem om de två aktörerna inte ingår i samma insatsgrupp eller om de inte ingår i två samtidiga insatser. Om exekveringen av aktiveringsplanerna däremot resulterar i en resurskonflikt, så kan detta emellertid få förödande konsekvenser för den gemensamma krishanteringsförmågan. Detta projekt behandlar frågan om hur konflikter i resursaktivering kan upptäckas genom en analys av resursaktiveringsplaner.

En grundläggande faktor för att resurskonflikter ska kunna uppstå är resurserna är begränsade i något avseende som exemplifierat ovan. I det svenska krishanteringssystemet finns ytterligare två faktorer som kan bidra till uppkomsten av resurskonflikter:

- Ansvariga krishanteringsorganisationer är självbestämmande och oberoende av varandra.
- Ansvariga krishanteringsorganisationer ska samverka för att hantera en uppkommen kris.

Vi har utgått från att en krishanteringsorganisation därmed oberoende av andra krishanteringsorganisationer utformar sina resursaktiveringsplaner. Eftersom krishanteringsorganisationerna förväntas samverka, kan det vid en kris uppstå beroenden mellan dessa aktiveringsplaner. Sådana beroenden kan orsaka konflikter i resursaktivering och på så vis verka negativt på den gemensamma krishanteringsförmågan; t.ex. då två organisationer har avtalat med samma (eller med olika) resursleverantör(er) om en och samma resurs. Konflikter i resursaktivering behöver inte uppstå enbart från samtidiga behov, utan de kan även ha sin orsak i allt från felskrivna avtal till alltför kort tid mellan två användningar av samma resurs.

Huvudresultatet från projektet är en metod för att möjliggöra upptäckt av potentiella konflikter i resursaktivering som kan uppstå då två eller flera oberoende aktörer exekverar sina resursaktiveringsplaner. Ansatsen utgår från att resurser aktiveras utifrån exekvering av avtal mellan resursbehövare och resursägare. Dessa avtal utgör en decentralisering av resursaktiveringen och vi antar att de inblandade aktörerna inte har full insyn i hur andra avtal än de som de själva har ingått har utformats. Antagandet avspeglar dels oberoendet mellan myndigheter och dels de privata aktörernas möjlighet att allokera resurser utifrån målsättningar som inte har direkt koppling till krishantering. T.ex. kan en privat aktör ha överallokerat sina resurser i vinstmaximerande syften. Vi säger att två planer som gör anspråk på resurser på oförenliga sätt, t.ex. genom att ge upphov till resursaktiveringskonflikter, är *inkonsistenta*.

Problemet med att hitta inkonsistenser i avtal är ingalunda enkelt, vilket berörs nedan i avsnittet Problemets komplexitet. För att ge en uppfattning redan nu, kan sägas att om vi har 100 avtal om resursaktivering, så kan dessa, i det generella fallet, genomföras på k^{100} olika sätt, där k är ett tal som beror (bland annat) på hur lång tid aktörerna har på sig att exekvera avtalen, och i det generella fallet är minst lika stort som antalet avtal som analyseras. I fallet med 100 avtal finns det alltså minst 100^{100} möjliga kombinationer av exekveringar av avtalen, vilket är större än det uppskattade antalet atomer i universum. Innebörden av den stora siffran är att redan för relativt små aggregat av resursaktiveringsavtal, så blir antalet möjliga exekveringar av dessa väldigt stort. För att hitta alla inkonsistenser i avtalen, så behöver vi i det generella fallet analysera alla möjliga exekveringar av avtalen. Problemet behöver delas upp i flera beståndsdelar och förenklas genom flera antaganden. I en verklig tillämpning av kalkylen kommer man att kunna reducera komplexiteten på problemet bland annat genom att utesluta vissa sekvenser av exekveringar av avtalen.

Inkonsistenser i resursaktiveringsplaner kan bero på flera faktorer. Projektet har tittat på en signifikant och grundläggande mängd sådana faktorer. Specifikt har vi behandlat problem som kan härröras till allokeringsavtal för, och tillgänglighet av, resurser. Det finns ytterligare faktorer som kan ge upphov till resursaktiveringskonflikter, vilka kan härröras till de mekanismer som kopplar resurser till behov och även till mekanismer som kan användas för att styra dessa kopplingar. För ett antal behov, kan åtkomsten till en resurs t.ex. styras genom prioritering. Prioriteringar lägger till regler till de befintliga kopplingsmekanismerna för att minska eller öka tillgången till resurser på basis av vilket behov som ger upphov till resursförfrågan. Om man tar problemets komplexitet i beaktning, så förefaller det dock inte vare möjligt att formulera fungerande prioriteringar och delningsprinciper annat än för begränsade delar av de gemensamma aktiveringsplanerna. För att kunna adressera problemet behövs först en överblick av omständigheter under vilka resurskonflikter kan uppstå, och också en förståelse av konfliktens karaktär så att adekvata åtgärder kan sättas in. Ytterligare en faktor som kan ge upphov till resurskonflikter som vi inte i någon större utsträckning har adresserat i projektet är fysiska egenskaper hos resurserna och villkor som uppstår då man tar mer verklighetstrogna representationer av resurser och insatser i beaktning. Att ta sådana aspekter i beaktning skulle dels kunna ge en mer finkorning

analys av vilka konflikter som kan uppstå mellan aktiveringsplaner, dels skulle det kunna ge underlag för att tolka de konflikter som upptäcks av kalkylen som tagits fram i projektet. En möjlig fortsatt utveckling av föreliggande ansats skulle kunna vara att studera hur faktorer som finkornighet i verklighetsmodellen, prioriteringar och delningsavtal kan påverkar förekomsten av inkonsistenser i aktiveringsavtal.

Projektet har resulterat i en kalkyl och en prototypimplementering, med vilka potentiella resursaktiveringskonflikter kan upptäckas, baserat på befintliga allokeringsavtal och resurstillgänglighet.

Baserat på komplexiteten i problemet och betydelsen av att kunna förutse resurskonflikter i samtidiga insatser är utvecklingen av ett automatiserat verktyg för att hitta och klassificera möjliga resursaktiveringskonflikter angelägen. Den kalkyl och prototyp som har tagits fram i projektet kan utgöra ett första steg mot ett sådant verktyg.

1.1 Avgränsningar av begrepp och ansats

Ansatsen fokuserar på resursaktivering. Vi gör ingen skillnad på resursaktivering i samband med kris eller med vardagshändelser. I själva verket skulle man kunna säga att vi tittar på resursaktivering i samband med ett behov från samhällets sida att upprätthålla eller uppnå ett visst *tillstånd* (t.ex. att en isbelagd väg ska göras farbar genom halkbekämpning). Ett tillstånd beskrivs som en tilldelning av värden på ett antal *omgivningsvariabler*. (Ett exempel på en omgivningsvariabel är snödjupet på en viss väg.)

Ett *insatsbehov* uppstår genom en *indikation* (även kallat *larm*) på att en omgivningsvariabel har antagit ett oacceptabelt värde. Vad som anses som oacceptabelt kan definieras från fall till fall. I scenariot (avsnitt 1.2) har vi t.ex. angett att ett snödjup på 6 cm på en väg är oacceptabelt (vilket föranleder en snöbekämpningsinsats).

Vi tar för givet (d.v.s. vi anser det ligga utom projektets ramar) att det finns föreskrivet vilken typ av insats som krävs för att uppfylla ett givet insatsbehov. Vi tar också för givet att det finns föreskrivet vilka resurser som behövs för en viss insats och för hur dessa resurser ska dimensioneras utifrån insatsens omfattning. När det gäller sammansatta resurser tar vi för givet att sammansättningen är given på förhand. Vid en tillämpning av den framtagna kalkylen på ett verkligt fall, måste dessa antaganden ses över.

Ett insatsbehov triggar utförandet av en *insats* genom exekvering av en *insatsplan*. Vi säger att det finns en *insatsansvarig* som har ansvar för att insatsen utförs. Den insatsansvarige kan delegera utförandet av insatsen till en eller flera *insatsutförare* (t.ex. genom att lägga ut uppgiften på entreprenad).

Insatsutföraren har ansvar för att sätta samman en *insatsgrupp*, vilket görs genom att denne exekverar en *formeringsplan*. I det allmänna fallet innefattar det att aktivera förmågor, tjänster och resurser. I projektet är formering begränsad till att aktivera ett antal resurser. Resurserna som ska aktiveras kan

vara sammansatta av flera komponenter, vilka i sin tur kan behöva aktiveras genom ytterligare formeringsplaner. Då insatsutföraren har aktiverat alla resurser som behövs, genomförs insatsen. Efter genomförd insats, återställs alla resurser och delresurser till sina respektive ägare. En formeringsplan utgörs av ett antal resursallokeringar, vilka är formulerade som avtal mellan resursbehövare och resursägare.

I begreppet *krishanteringsorganisation* inkluderas myndigheter och privata aktörer som har fått i uppdrag att tillhandahålla tjänster och resurser som behövs vid en krisbekämpning. Uppdragen beskrivs i vår ansats som avtal mellan aktörerna.

Begreppet *beredskapsplan* används informellt i rapporten. Vad som avses är de avtal som insatsansvariga och insatsutförare enligt ovan har ingått för att aktivera de resurser som behövs i en insats.

1.2 Scenario

Scenariot som är förprogrammerat i prototypen och som hänvisas till i rapporten rör en liten fiktiv kommun i Sverige under ett dygn under en längre period där vägarna blir ofarbara både på grund av isbeläggning och ihållande snöfall. Scenariot är stiliserat och saknar flera aspekter av realism, men det är tillräckligt komplext för att illustrera den framtagna kalkylen på ett effektivt sätt. De resurser som vi tittar på är de som behövs vid bekämpning av halka och snötäcken på vägarna i staden med omnejd.

Vägarna i staden är tio till antalet och de har fyra s.k. omgivningsvariabler: *snödjup*, *halka, snövallsbildning*., och *förekomst av person vid väg som behöver medicinsk tillsyn*. Snödjupet mäts i centimeter och det ökar med en normalfördelning kring 3 cm i timmen då snön faller, vilket i sig styrs av en slumpprocess (Markov-kedja) som genererar ihållande snöfall med korta perioder (runt 30 minuter och ibland upp till två timmar) av uppehåll. Halka är en binär slumpvariabel och den slår till på en väg runt fyra timmar efter att vägen har blivit halkbekämpad. Då det har snöat tillräckligt mycket (sammanlagt 35 cm) (och oberoende av ifall en väg har blivit snöbekämpad), anses snövallar ha bildats.

Under det aktuella dygnet är det flera tjänster som ska utföras. Dessa är:

- ordinarie snöbekämpning på vägar vars snödjup överstiger 6 cm
- ordinarie halkbekämpning på vägar där halka har uppstått
- ett flertal ambulansutryckningar då stadsinnevånare har halkat eller blivit nedkylda. Ambulansutryckning fordrar samtidig halk- och/eller snöbekämpning
- en akut ambulanstransport genom staden till länssjukhuset i en närliggande stad, vid något tillfälle under dygnet (styrs av en slumpvariabel). Detta fordrar att färdvägen halk- och snöbekämpas

- utlåning av snöbekämpningsresurser till länsstyrelse under två timmar vid ett-tiden på dagen (styrs av en slumpvariabel)
- diverse transporter av förnödenheter och bränsle till stadens omnejder
- bortforslande av snövallar där de har bildats under dygnet.

Kommunen ansvarar för den ordinarie snö-, halk- och vallbekämpningen, vilka den har lagt ut på entreprenad till ett antal olika privata företag. En snöbekämpningsinsats kräver 1 plog, vilken i sin tur kräver 1 förare och 1 enhet bränsle (vilket ska motsvara en viss volym av bränslet). De privata företagen har oberoende av varandra slutit avtal med förar- och bränsleleverantörer. Efter avslutad ploginsats återlämnas föraren medan bränslet är förbrukat. Halkbekämpning kräver två lastbilar vilka i sin tur kräver förare, bränsle och salt, samt eventuellt en plog ifall snödjupet överskrider en viss nivå. För att forsla bort snövallar krävs en lastare och två lastbilar, och därmed tre förare och tre enheter bränsle. Snö-, halk- och vallbekämpning måste vara avslutade inom tre timmar efter att respektive behov har uppstått. Den tid det tar för att snö- eller halkbekämpa en väg varierar enligt en normalfördelning kring en timme, medan vallbekämpning tar kring två timmar att utföra.

Den medicinskt ansvariga myndigheten har avtal med halkbekämpnings-aktörer, vilka sin tur har avtal med snöbekämpningsaktörer för att möjliggöra utryckning med ambulans vid behov.

En privat aktör har avtal med en halkbekämpningsaktör och en snöbekämpningsaktör för att kunna halk- och snöbekämpa den väg där den akuta ambulanstransporten far genom staden på väg till länssjukhuset.

Kommunen har ingått avtal med länsstyrelsen om att låna ut två plogar vid behov från länsstyrelsen. Dessa plogar rekvireras från kommunens ordinarie plogentreprenörer.

Transporter av förnödenheter och bränsle utförs av privata företag, vilka har ingått avtal med halk- och snöbekämpare för att kunna genomföra transporter vid behov.

Scenariot drivs genom ett antal larm som uppstår under scenariots dygn. Det finns ett larm för varje tjänst som måste utföras:

1. larm för att snönivån har överstigit 6 cm på en väg.
   (Detta larm reagerar kommunen på och exekverar ett av sina avtal där ansvaret för snöbekämpning har lagts ut på entreprenad. Om ingen av de fem entreprenörerna kan leverera snöbekämpningstjänst för att de redan är ute och snöbekämpar på uppdrag av kommunen, så sägs kommunen ha brutit sitt avtal om snöbekämpning.
2. larm för att en väg har fått isbeläggning.
3. larm för snövallar
4. larm för att en transport av förnödenheter eller bränsle måste genomföras

5. larm för att länsstyrelsen vill låna plogar av kommunen
6. larm för att ambulansutryckning
7. larm för att en akut ambulanstransport måste genomfara staden

Vid ett larm, sätts den insatsansvariges insatsplan i verket, vilket innebär att den ansvarige exekverar ett av sina avtal där ansvaret för att utföra insatsen delegerats till en entreprenör. Entreprenörens formeringsplan sätts därpå i verket. Schematiskt ser det ut som följer:

1. Larm L uppstår. L tas emot av en insatsansvarig A.
2. A har en insatsplan, där det är specificerat vilken insats O som måste genomföras, samt vilka entreprenörer som A har slutit avtal med för att genomföra O.
3. A väljer en av de tillgängliga entreprenörerna, E, och binder E till O och larmet L. Om det inte finns någon entreprenör tillgänglig att utföra O, så är det något avtal som har blivit brutet. Beroende på omständigheterna, så är det A själv eller någon av entreprenörerna som anses ha brutit ett avtal.
4. Om en entreprenör E finns tillgänglig, så genomför E sin formeringsplan för att utföra O. Detta involverar att sätta samman en insatsgrupp som är kapabel att utföra O, vilket i sin tur innebär att rekvirera de resurser som behövs för att utföra O. Resurser kan behöva rekvireras i flera led vilka kan involvera att ytterligare entreprenörer exekverar sina formeringsplaner.
5. När E har erhållit alla resurser och insatsgruppen har formerats, så genomförs O.
6. Efter att O har genomförts, så avvecklas den insatsgruppen
7. Till sist avgörs om O var genomförd i tid och i enlighet med övriga krav som ställts på utförandet.

Ett antagande i scenariot är att de insatsansvariga inte nödvändigtvis har anpassat sina beredskapsplaner till varandras. I värsta fall uppstår deras behov samtidigt och de har kontrakterat exakt samma entreprenörer för att utföra respektive operationerna. Det kan också vara så att de larmade aktörerna har anlitat olika utförare, men att utförarna i sin tur har anlitat samma resursleverantörer, antingen direkt (e.g. att två snöbekämpare har anlitat samma plog) eller i flera led (e.g. att två snöbekämpare har anlitat två olika plogar, men att dessa i sin tur har anlitat samma förare).

Ett andra antagande i scenariot är att formeringsansvariga inte har tillgång till kunskap om hur ägare av sammansatta resurser har slutit avtal med underleverantörer av resurser.

1.3 Problemställning

Den övergripande frågeställningen för projektet rör den gemensamma förmågan hos krishanteringsorganisationer att hantera aktivering av gemensamma begränsade resurser i samband med kris.

Frågeställningen kring konsistens i resursaktiveringsplaner utgör en specificering av den övergripande formuleringen av problemet. Den innefattar också problematik kring den ursprungliga ansatsen till problemet, vilken bestod i en formalisering av rättighetshantering för resursaktivering, baserat på en utvidgning av en existerande kalkyl för informationsåtkomst [7, 8]. Specificeringen av frågeställningen till konsistens i aktiveringsplaner möjliggör en analys av orsakerna till att resursåtkomst överhuvudtaget behöver regleras. Ansatsen i projektet utgår från att resurser aktiveras i enlighet med ingångna avtal mellan resursägare och resursbehövare, vilket innebär att formaliseringen av styrningsmöjligheterna för resursåtkomst redan till stor del är bestämd.

1.3.1 Specifik problemställning

I det här avsnittet beskrivs den specifika problemställning som projektet har adresserat. Det är denna problemställning som kalkylen är avsedd att kunna hantera. Kalkylen ger svar på frågorna 1 och 2.

Förutsättning 1 Vi antar först att det finns ett antal resurser som behövs i en eller flera insatser. Dessa resurser har en eller flera begränsningar, vilka exempelvis kan vara i antal; i volym; i räckvidd; i uthållighet; eller i konfigurabilitet. Resurserna kan vara sammansatta och komponenterna kan komma från flera olika leverantörer. Leverantörerna i sig kan också utgöra en begränsning: de kan ha överallokerat sina resurser; de kan vara mer eller mindre pålitliga; eller de kan följa arbetstider som inte passar en insats.

Förutsättning 2 Den andra förutsättningen är att det finns ett antal behov av dessa resurser. Dessa behov uppstår då en ansvarig krishanteringsorganisation genomför en insats. Behoven aktualiseras i och med att insatsplanerna och formeringsplanerna exekveras.

Förutsättning 3 Aktivering av resurser utifrån behov styrs av ett antal resursaktiveringsavtal. Resursaktiveringsavtalen reglerar hur de befintliga resurserna kopplas till behoven. Det sker genom allokering och förfrågan. Resursaktiveringsavtalen har ingåtts oberoende av varandra.

Fråga 1 Givet en kombination av resursbehov, kan de befintliga resurserna täcka alla dessa behov? Denna fråga förbiser resursaktiveringsavtalen och utgör en kontrollfråga. Ett nekande svar här betyder ett det inte finns tillräckligt mycket resurser att tillgå hur än resursaktiveringsplanerna utformas.

Fråga 2 Givet ett antal avtal om resursaktivering, och en given kombination av resursbehov, kan det uppstå konflikter i aktiveringen av befintliga resurser då resursaktiveringsavtalen exekveras och efterlevs? Ett jakande svar skulle kunna tyda på att resursaktiveringsavtalen bör ses över.

I avsnitt 3 ges ett förslag på en utökning av kalkylen så att fler typer av frågor kan besvaras.

1.4 Kalkylens potentiella nytta

Resursaktiveringsproblem är en viktig aspekt av kriser. Om resurser inte aktiveras då de ska kan vardagshändelser eskalera till kriser och kriser kan förvärras. Projektet har tittat på en specifik aspekt av resursaktivering, och därigenom tagit fram en kalkyl som kan upptäcka potentiella problem i resursaktiveringsplaner. Kalkylen är framtagen i syfte att kunna vara till nytta för aktörer i det svenska krishanteringssystemet. Nedan följer fyra exempel på hur kalkylen skulle kunna användas i praktiken.

Ömsesidig översyn av beredskapsplaner
Två, eller flera, aktörer med överlappande eller närliggande ansvarsområden, t.ex. geografiskt, skulle gemensamt kunna använda kalkylen för att ta reda på om och hur deras resursaktiveringsplaner kan komma att störa varandras krisberedskapsplaner.

Iterativ interaktiv förbättring av beredskapsplaner
Kalkylen kan användas för att hitta svagheter i befintliga resursaktiverings-planer. Vid upptäckt av en svaghet kan denna eventuellt avhjälpas med tillägg eller omformulering av befintliga resursaktiveringsplaner. Kalkylen kan indikera om avtal för reservresurser kan vara nödvändiga att ta i beaktning.

Placering och dimensionering av resurser
Genom att mata in representationer av befintliga resurser och aktiveringsplaner i prototypimplementeringen, kan en ansvarig aktör köra ett antal scenarier för ett upptäcka under vilka omständigheter resursbrister kan uppstå. Utifrån kalkylens resultat kan samma scenarier köras med alternativa placeringar och dimensioneringar av befintliga resurser, för att på så vis kunna optimera placering och dimensionering av dessa.

Uppföljning och analys av insatser
Om en given insats visat sig vara behäftad med resursaktiveringsproblem, så kan kalkylen vara till hjälp för att se om dessa problem hade sina ursprung i aktiveringsplanerna.

1.5 Begränsningar i kalkylen

Både den framtagna kalkylen och ansatsen har ett antal begränsningar, vilka är viktiga att klargöra.

Att det generella problemet är komplext medför att varje tillämpning av kalkylen på en given domän potentiellt kommer att ha sin egna specifika uppsättning förenklande antaganden. Tillämpningen av kalkylen med en viss uppsättning antaganden på en domän kan visa sig vara effektiv, medan samma antagande kanske inte alls gäller eller ger effektiva resultat i en annan domän.

Kalkylen ger en indikation på konflikter i resursaktivering med avseende på exekvering av de planer för resursaktivering som analyseras, och inte på andra möjliga problem som kan föreligga i beredskapsplaner och krishanterings-

organisationer. Juridiska problem, eller kulturella problem i krishanterings-organisationer är exempel på problem som inte direkt fångas upp av föreliggande kalkyl.

Vidare finns det flera synbarliga tillkortakommanden med en ansats baserad på avtal. Dessa behöver dock i praktiken inte reflektera problem vare sig med avtal eller med kalkylen, eftersom de pekar på problem som gäller generellt för krishantering [4]. Tvärtom, kalkylen är framtagen för att kunna användas för att upptäcka, analysera och avhjälpa dessa problem:

- Att avtal för resursaktivering efterlevs är inte en garanti för att en kris ska kunna avhjälpas med de aktiverade resurserna (de kan vara trasiga, felkonfigurerade, eller liknande).
- Att avtal för resursaktivering är ingångna är inte en garanti för att resurserna faktiskt aktiveras (avtal kan t.ex. brytas)
- Avtal för resursaktivering kan ha ingåtts på felaktiga grunder, eller ha baserats på felaktiga modeller av vilka resurser som behövs vid en insats.
- Kriser kan utvecklas så att de avtal om resurser som har ingåtts inte ger tillräckligt med resurser för en insats.
- I krisbekämpning krävs ofta att aktörer kan aktivera resurser med omedelbar verkan baserat på aktörens erfarenhet och dennas direkta kontakt med krisen. En avtalsbaserad ansats får inte komma i vägen för improviserade åtgärder genom att t.ex. kräva långdragna omförhandlingar om aktiveringsavtalens innehåll.
- En lyckad insats beror inte enbart på att resurser har kunnat aktiveras på ett tillfredställande sätt. För en fungerande krishanteringsorganisation krävs att flera andra aspekter av krishantering också fungerar: t.ex. kommunikation, ledarskap och tillit.

1.6 Metod och Tillvägagångssätt

Arbetet har berört flera vitt skilda områden och det har varit nödvändigt att inhämta kunskap både från krishanteringssammanhang och från formella metoder, samt att förena dessa områden till en kalkyl som både är relevant för det svenska krishanteringssystemet och är adekvat i en formell mening.

Metoden har i huvudsak varit explorativ och inkrementell. Målsättningen med arbetet har varit att:

1. Bygga en matematisk modell för
   - en begränsad del av beredskapsplaner i termer av avtal,
   - resurser som används i samband med kris,
   - resursbehov, och

    - procedurer kring allokering och aktivering av resurser
2. Utveckla en algoritm för att upptäcka konflikter i resursaktivering
3. Implementera algoritmen i form av en prototypkalkyl

Validering av modellen och algoritmen har bestått i utvärdering av dessa genom (i) jämförelse med inhämtad kunskap om det svenska krishanteringssystemet; och (ii) exponering av ansats genom publikationer och presentationer på dels konferenser och workshops, dels i samband med mer informella möten både med krishanteringspersonal och personer med bakgrund i formella metoder. Implementeringen har validerats genom felsökning och testkörningar av prototypen.

Prototypen har utvecklats i programmeringsspråket Python [5], och använder sig i huvudsak av simuleringsmodulen SimPy [6] både för att simulera resursbehov, resursaktivering och exekvering av (den formella representationen av) beredskapsplaner. I takt med att algoritmen tagit form och prototypen har mognat, har Python-koden i allt större utsträckning intagit platsen som den formella modellen.

Under den största delen av projekttiden var arbetet i huvudsak explorativt med syfte att förstå problemet både från ett modelleringsperspektiv och från ett krishanteringsperspektiv, samt att få en överblick över vilka formaliseringsmöjligheter som stod till buds. Implementeringen av prototypen utgjorde en viktig komponent i arbetet med att förstå problemet. En viktig avvägning var mellan den realism som tycktes erfordras från krishanteringssammanhang och lämplig abstraktionsnivå för den matematiska modelleringen.

Kontraktspråket är till stora delar influerat av, och bygger på, en formalism utvecklat Ralph-Johan Back och medarbetare [3]. Målsättningen i Ralph-Johan Back och medarbetares arbete syftar till att verifiera egenskaper hos dataprogram, vilket är helt skilt från syftet med användningen av kontraktsformalismen här. Semantiken för det i projektet använda kontraktspråket utgår till viss del från Ralph-Johan Backs och medarbetares arbete, men skiljer sig i sin nuvarande form från deras semantik. Semantiken för föreliggande kontraktspråk bygger snarare på intuition kring vad gäller för krishanteringssammanhang, och är integrerat i prototypen som metoder vilka definierar exekveringen av termer och protokoll.

Algoritmen bygger på simulering. Detta var ett val som föreföll naturligt eftersom det generella problemet beräkningsmässigt är mycket komplext (minst exponentiellt i antalet kontrakt). Problemet har under projektets gång förenklats på flera sätt, vilket har minskat den beräkningsmässiga komplexiteten. Det är möjligt att det finns en exakt lösning på det förenklade problemet, men inga försök har gjorts för att hitta en sådan.

2. Korsvis Analys av Konsistens i Avtal

Projektets huvudresultat är en kalkyl som har fått namnet Korsvis Analys av Konsistens i Avtal. Kalkylen är avsedd för att möjliggöra upptäckt av resursaktiveringskonflikter som kan uppstå då flera avtal om resursaktivering exekveras samtidigt. Det här avsnittet beskriver kalkylen på ett informellt och förhoppningsvis tillgängligt sätt.

2.1 Grundläggande begrepp

I detta avsnitt definieras de flesta av de begrepp som användas hädanefter i rapporten. Övriga begrepp definieras i takt med att de introduceras.

2.1.1 Omvärldsvariabler

För att modellera omvärlden använder vi oss av s.k. *omvärldsvariabler*. En omvärldsvariabel har ett namn och ett antal möjliga värden. Omvärldsvariabler kan grupperas och forma en namngiven s.k. *infrastruktur*. Exempel på infrastrukturer är de namngivna vägarna i den fiktiva kommunen i prototypscenariot. De har omvärldsvariablerna *snödjup*, *isbeläggning*, *snövallar*, och *förekomst av person i behov av medicinsk tillsyn*. Även mängden av vägar är en omvärldsvariabel. Exakt vilka omvärldsvariabler som förekommer i ett scenario bestäms från scenario till scenario och beror på vad det är man vill modellera.

Här följer några exempel på omvärldsvariabler som förekommer i prototypimplementeringen av scenariot. Scenariot har en *global klocka* och varje avtal har en *lokal klocka*. Omvärldsvariabeln *global klocka* används bland annat för att styra en stokastisk process som styr omvärldsvariabeln *snöfall*. Det finns en omvärldsvariabel som är en lista av entreprenörer som har fått i uppdrag av kommunen att utföra snöbekämpning på vägar som är igensnöade. Den lokala klockan i ett avtal används bland annat för att kunna avgöra om en snöbekämpningsinsats har blivit utförd i tid. Varje resurs har en omvärlds-variabel som anger om resursen är tillgänglig eller inte.

2.1.2 Termer, Protokoll och Avtal

Omvärldsvariablernas värden kan bland annat uppdateras, läsas av och testas mot ett villkor. En del av dem uppdateras automatisk av regler som har angivits i scenariot, t.ex. att snödjupet på en väg ökar när det snöar. Andra uppdateras genom exekvering av termer i protokoll eller avtal.

En *term* är en instruktion om att uppdatera, läsa av eller testa en omvärldsvariabel. En *aktör* kan kopplas till en term (men alla termer behöver inte ha en aktör kopplad till sig). Om en aktör är kopplad till en uppdatering sägs denne vara ansvarig för termens utförande genom att uppdatera en

omvärldsvariabel enligt termens instruktion. T.ex. finns det en term, `delegate`, för delegering av insatser. Vid utförande av termen `delegate(kommun,sbek,beppe)` uppdateras omvärldsvariabeln (av typ lista) som innehåller alla aktörer till vilka kommunen har lagt ut snöbekämpning (`sbek`) på entreprenad genom att lägga `beppe` till listan. I avsnitt 2.4 nedan beskrivs termerna mer ingående.

Ett *protokoll* är en följd av termer. Antag att *x*, *y*, och *z* är termer och att aktörerna *a, b*, och *c* är kopplade till *x, y*, respektive *z* (eller med andra ord: har ansvar för *x, y*, respektive *z*). Då är följden (*x,y,z*) ett exempel på ett protokoll. Vidare sägs *a, b*, och *c* ha ingått ett *avtal* att genomföra protokollet (*x,y,z*). Om *b*, säg, fallerar att utföra instruktionen i *y*, då sägs *b* ha brutit avtalet (*x,y,z*) och aktörerna *a* och *c* sägs vara befriade från (*x,y,z*) (givet dock att varken *a* eller *c* är identisk med *b*).

En *körning* (eller *exekvering*) av ett protokoll *P* består av en tilldelning av en *duration* och en *starttid* för varje term i *P*. Om *x* och *y* är termer i *P* och *x* kommer före *y* i följden av termer som utgör *P*, då måste starttiden för *x* tidsmässigt vara före starttiden för *y*. I princip kan durationen för *x* överlappa durationen för *y*, men för att förenkla algoritmen så har vi infört antagandet att instruktionen i *x* måste vara avslutad innan *y* kan börja.

Det finns ytterligare en uppsättning omvärldsvariabler som spelar en viktig roll i prototypen. Nämligen protokollens *interna variabler*. Ett givet protokolls omvärldsvariabler är inte åtkomliga för andra aktörer än de som kopplade till någon term i protokollet. Det finns dock termer som instruerar kommunikation av de interna variablernas värden till andra protokoll. Genom att utnyttja de interna variablerna och kommunikationsmekanismerna mellan protokoll, kan t.ex kommunen knyta en viss snöbekämpningsinsats till bara en av de entreprenörer som den har avtalat med.

2.1.3 Resurser, Konfigurationer och Insatser

En *resursinstans* består av ett *namn* och ett antal *attribut*. Namnet är unikt och fungerar som identifikation av resursen. Exempel på attribut är *typ, geografisk position, ägare, tillstånd* (e.g. *tillgänglig* eller *upptagen*), *kapacitet*, eller *ackumulerad arbetstid*. Vilka attribut en resursinstans har anges i en *resursmodell* och kan varieras efter en given tillämpnings förutsättningar. I resursmodellen anges också på vilket sätt en sammansatt resursinstans är uppbyggd, samt hur resursinstansen förbrukas eller på annat sätt påverkas av användning.

En *konfiguration* är en beskrivning av resursinstanser vilken anger typ och antal samt en beskrivning av attributen hos instanserna. Exempelvis kan en konfiguration ange `typ=plog` och `antal=2` med ytterligare attribut som beskriver t.ex. var dessa denna plog ska befinna sig vid en viss tidpunkt och hur länge de ska operera inom ett visst område. Kalkylen gör det möjligt att med hjälp av attribut beskriva resursinstanser med stor noggrannhet.

Vi skiljer resursinstanser från resurser. En *resurs* beskrivs med hjälp av en konfiguration och den har en *ägare*. Ägaren kan på så vis erbjuda resursinstanser i enlighet med resursens konfiguration. Notera att ägaren av en

resurs inte behöver äga några resursinstanser (för vilka denne i så fall ingår avtal med instansägare för att erhålla). Ett exempel: resursägaren A kan erbjuda en resursen `vakt` (dvs en människa som vaktar något) med 24 timmars uthållighet, om A har tillgång till, säg, tre resursinstanser `vakt` vilka vardera arbetar 8 timmar i sträck. En sådan konfiguration skulle kunna se ut som `(typ=vakt, antal=1,uthållighet=24)`. Det gäller att A har slutit avtal med de tre vaktinstanserna på ett sådant sätt att A kan uppfylla ett löfte om att kunna leverera enligt sin konfiguration. Ett till exempel: om B är en resursägare med konfiguration (typ=plog, antal=1), så kan B allokera en plog till flera aktörer i behov av en plog. Det gäller bara att B har slutit avtal med tillräckligt många ploginstansägare, för att kunna uppfylla alla sina löften om att leverera en plog (annars sägs B ha överallokerat sin resurs).

För begreppet *insats* har vi antagit en modell. En insats har ett namn (t.ex. halkbekämpning) och ett antal attribut: *prioritet*, *duration*, *resurser*, och *effekt*.

Attributet prioritet anger två värden: *nominell duration* och *prioritetsvärde*. Den nominella durationen är ett värde som används i avtal om hur lång tid en insats får ta. Per default är detta 3 timmar, men värdet kan specialiseras till insatsen och vem som är insatsansvarig. Prioritetsvärdet har ingen effekt i nuvarande implementering av prototypen.

Attributet duration är den modellerade faktiska tiden det tar för att genomföra insatsen. Det är en normalfördelat slumpvariabel med medelvärde 1 timme. Durationen kan också anpassas till vem som är insatsansvarig.

Attributet resurs anger vilka resurser som behövs för att utföra insatsen. Vi antar i projektet att det finns beskrivet vilka resurser som behövs och hur de ska dimensioneras för en given insats.

Attributet effekt tar en omvärldsvariabel som argument och anger vilken effekt insatsen har på denna omvärldsvariabel givet dess nuvarande värde. T.ex., utförande av snöbekämpning på en väg gör att snödjupet på vägen blir noll i prototypen. Detta är naturligtvis naivt, men om man vill är det redan förberett i prototypen att anpassa insatsmodellen till mer realistiska värden.

2.2 Modell för Krishantering

Krishanteringsmodellen i projektet är tjänstebaserad och har hämtat influenser bland annat från STIL-ramverket [2] och Göransson et al [1]. En grundläggande idé är att en krishanteringsinsats genomförs av tillfälligt sammansatta organisationer speciellt anpassad för den kris som ska bekämpas. Efter att krisen har blivit bekämpad, avvecklas den tillfälligt sammansatta organisationen.

I vår modell står resursbehov och resursaktivering i fokus. Resursbehov uppstår som en effekt av ett *insatsbehov*. Vi säger att ett insatsbehov uppstår när en eller flera *omvärldsvariabler* antar oacceptabla värden. T.ex., uppstår ett behov av halkbekämpning då en väg har fått en isbeläggning, eller med

andra ord, att framkomligheten och säkerheten på en väg har antagit oacceptabla värden på grund av isbeläggning.

Bland omvärldsvariablerna finns det tre typer som används för att modellera uppkomsten av insatsbehov. Först har vi *generatorer*, vilka dynamisk förändras enligt fördefinierade regler. Dessa regler kan konfigureras enligt de förutsättningar som gäller för en given tillämpning. Ytterligare en uppsättning regler bestämmer hur omvärldsvariabler kopplade till infrastrukturer påverkas av hur värdena på generatorerna förändras (t.ex. hur snödjupet på en väg förändras som ett resultat av värdet på generatorn *snöfall*.) Den tredje urskilda typen av omvärldsvariabler kallas *indikatorer*. En indikator talar om ifall en viss omvärldsvariabel som hör till en viss infrastruktur har antagit ett s.k. *oacceptabelt värde*. När en indikator definieras, kan man också specificera vilka värden på en eller flera omvärldsvariabler som ska uppfattas som oacceptabla. I prototypscenariot har det t.ex. angetts att ett snödjup på över 6 cm på en väg är oacceptabelt.

Då en indikator visar på att en omvärldsvariabel har antagit ett oacceptabelt värde produceras ett *larm*. Ett larm innehåller den information som är nödvändig för att kunna utföra en insats för att återställa omvärldsvariabel som har antagit ett oacceptabelt värde (t.ex. information om vilken väg som behöver plogas, om omvärldsvariabeln var snödjup).

I prototypen kan man definiera s.k. *larmavtal*. I ett larmavtal definieras en indikator och de värden på angivna omvärldsvariabler som ska ge upphov till larm. Det ger flexibilitet att förändra insatsförutsättningarna under körning av scenariot. T.ex., skulle man kunna vilja modellera en förändring av prioriteten för snöröjning på vissa vägar att endast initieras om snödjupet når 12 cm om det snöar mycket och resurserna är ansträngda.

I vår modell behöver man ange en aktör, kallad *larmansvarig*, som är ansvarig för att avlyssna larm och sända de vidare till den aktör som är ansvarig den insats som enligt larmet erfordras. I framtida tillämpningar av kalkylen kan detta användas för att modellera begränsningar i larmhanteringskapacitet.

I ett s.k. *tjänstekontrakt* anges en aktör, kallad *insatsansvarig*, som är ansvarig för att

- Lyssna på en eller flera larmansvarigas larm
- Initiera en insats för att svara på ett larmet
- Antingen genomföra insatsen själv eller delegera insatsen till någon som denna har ingått ett *delegeringsavtal* med (se nedan)

Tjänsteavtalet anger de villkor som ska gälla för utförandet av insatsen, t.ex. hur lång tid den ska få ta.

Ett *delegeringsavtal* upprättas mellan två aktörer så att en insats som den förste aktören är ansvarig för kan utföras av den andre. I prototypen kan man ange de omständigheter under vilka delegeringen gälla (t.ex. start- och sluttid, eller något villkor på en omvärldsvariabel). Insatsansvarig kan ingå delegeringsavtal med flera aktörer.

När insatsansvarig har kopplat en aktör, kallad *utförare*, till en insats, ska insatsen utföras. För att utföra insatsen behöver utföraren först sätta samman en insatsgrupp. Detta sker enligt en s.k. *formeringsplan*.

Vi antar i projektet att det finns beskrivet vilka resurser som behövs och hur de ska dimensioneras för en given insats. T.ex., för att forsla bort snövallar har vi antagit att det behövs en lastare och två lastbilar. Prototypen tillåter dock representation till godtycklig precision på sammansättning av insatsgrupper. För en verklig tillämpning på snöröjningsområdet finns det säkerligen flera olika sammansättningar som motsvarar flera olika krav på tjänsten.

Utifrån en sådan beskrivning definieras en *formeringsplan* som en samling avtal om aktivering av de resurser som behövs för en insats. Avtalen utgörs av allokeringsavtal mellan resursägare A och behövare B som säger att

- A förbinder sig att ställa resurser till förfogande enligt en konfiguration K,
- om och då B förfrågar i enlighet med konfigurationen K

I konfigurationen K anges alltid typ och antal av en resurs (i de fall antal är relevant, annars volym). Man kan också precisera K med ytterligare krav på den förfrågade resursen. T.ex. kan man ange *geografiskt läge* eller något annat som anses relevant för en given domän. Kalkylen tillåter även oprecisa uttryck av konfigurationsparametrar, som till exempel minst fyra stycken, eller längsta acceptabla ställtid. Däremot tar prototypen ännu inte hänsyn till sådana.

I formeringen av insatsgrupper kan några av de behövda resurserna vara sammansatta av flera komponenter. För att kunna aktivera en sammansatt resurs krävs att ägaren av denna exekverar sin formeringsplan för att aktivera komponenterna innan den sammansatta resursen kan aktiveras.

Då alla resurser som behövs har blivit aktiverade sätts insatsgruppen samman, och insatsen genomförs. Efter genomförd insats, avvecklas gruppen, och resurserna återlämnas till sina ägare.

Larmavtal, tjänsteavtal, delegeringsavtal och formeringsplaner kallas gemensamt för *resursaktiveringsplaner*, eller *aktiveringsplaner*.

2.3 Resursaktiveringsproblem

Kalkylen är avsedd för att hitta potentiella resursaktiveringsproblem som uppstår då flera aktiveringsplaner exekveras samtidigt. Det här avsnittet ger en sammanställning av de aktiveringsproblem som prototypen är implementerad att registrera. I avsnitt 3 ges exempel på andra aktiveringsproblem som skulle kunna implementeras i en fortsatt utveckling av kalkylen.

Låt A vara en aktör som förfrågar resursen R till en volym V1 från leverantören L. (Vi bortser här från det generella fallet med godtycklig konfiguration i allokeringsavtalen).

Konflikt i aktivering Om A har allokering för R hos L och L inte har tillräckligt mycket R för att tillgodose A:s förfrågan vid tidpunkten T, då sägs A:s förfrågan vara i konflikt med alla de förfrågander om R hos L vilka L tillgodoser vid tidpunkten T. L sägs ha överallokerat R. Detta kan vara en effekt av att någon av L:s underleverantörer inte kan leverera en komponent.

Restriktion i resursberoenden Om A har allokering för R hos L och L har kapacitet att tillgodose A:s förfrågan vid tidpunkten T, men inte kan göra det på grund av att en underleverantör till L ej kan tillgodose L:s behov vid tidpunkten T, då sägs A:s förfrågan inte kunna uppfyllas p.g.a. *restriktion i beroenden*.

Avsaknad av allokering Om A inte har en allokering för R, så avslås A:s förfrågan.

Allokering redan tillgodosedd Om A har en allokering för R hos L till en volym V, och A redan använder R till en volym W så att V-W < V1, då avslås A:s förfrågan. Detta kan inträffa också då A har ingått ett resursdelningsavtal om R med en annan aktör, C, och denne utnyttjar A:s allokering för R.

Utöver de ovan nämnda aktiveringsproblemen finns det även flera avtalsbrott som resulterar i problem med aktivering av resurser. T.ex.:

Avsaknad av delegeringsalternativ Om en insatsansvarig J inte kan delegera en insats p.g.a. att de aktörer som J har delegeringsavtal med redan utför insatser på uppdrag av J, då sägs insatsen vara ogenomförbar, och J bryter mot tjänsteavtalet. Om å andra sidan en av de aktörer M som J har delegeringsavtal med inte utför en insats på uppdrag av J en ändå inte kan utföra förfrågad insats, sägs också insatsen vara ogenomförbar, och J bryter tjänsteavtalet men i detta fall bryter även M delegeringsavtalet.

Utebliven förfrågan om resurs Om en resursförfrågan sker genom en term i ett komplext avtal, kan förfrågan om resursen utebli p.g.a. att avtalet är brutet redan innan resursfrågan har hunnit göras.

2.3.1 Ytterligare problem i resursaktivering

För följande problem finns det mesta som behövs för att kunna registrera dem redan implementerat. Dessa har inte blivit implementerade för att registreras antingen för att procedurerna för att upptäcka och identifiera dem förefaller att öka algoritmens komplexitet eller för att de inte är relevanta för nuvarande implementering av prototypen.

Restriktion i konfiguration (ökare av komplexitet) A har en allokering för R hos L och L har kapacitet att tillgodose A:s förfrågan vid tidpunkten T, så när som på ett antal attribut i förfrågankonfigurationen. Detta kan exempelvis inträffa då A vill ha R inom 60 minuter, men L kan bara leverera R inom, säg, 90 minuter. I prototypen avslås A:s förfrågan.

Restriktion i synkronisering Insatsgrupper sätts i nuvarande version av prototypen samman på nolltid (givet att resurserna kan aktiveras). I framtida versioner kan ställtid och andra faktorer som påverkar resursernas möjlighet att sättas in under givna tidsintervall tas i beaktning.

Restriktion i konfigurerbarhet Det är förberett i prototypen för att kunna ange förslitning och effektivitet hos resurserna som en funktion av tid och användning. De mekanismer som avgör om en resurs kan aktiveras känner dock inte igen sådana aspekter av konfigurationer ännu. (Det finns ännu heller inget interface för att ange dessa faktorer i samband med inmatning av ett scenario.)

2.4 Språk för avtal

I det här avsnittet ges en kort överblick av det språk i vilket resursaktiverings-avtalen skrivs. I slutet av avsnittet ges ett exempel på avtal hämtat från prototypen.

Det finns ett antal basala termer kallade: `assign`, `update`, `assert`, och `assume`. Avtal kan formuleras från dessa fyra typer av termer genom att ange en följd av instanser av dem.

Termen `assign` tar tre argument: *subject*, *variable*, och *value*. Innebörden av `assign`, eller vad som händer när av `assign` exekveras, är att variabeln *variable* sätts till värdet *value*. Om *value* är en lista av värden av samma typ som *variable* eller på annat sätt anges som ett aggregat av värden, så tilldelas *variable* ett av dessa värden slumpartat.

Termen `update` tar tre argument: *subject*, *variable*, och *updatefunction*. Innebörden av `update` är att *variable* tilldelas ett nytt värde enligt *updatefunction* som en funktion av variabeln gamla värde.

Termena `assert` och `assume` tar vardera två argument: *subject* och *expression*. Innebörden av assert är att uttrycket expression måste vara sant då termen exekveras. Exempelvis kan man kolla att en insats har blivit genomförd inom en viss tid genom att inkludera en assert-term i ett tjänstekontrakt:

`assert`(*insatsansvarig*, *local_clock* < *10*)

Om uttrycket *expression* inte håller, då sägs *subject* ha fallerat i utförande av termen (vilket i sig också kan leda till att *subject* sägs ha brutit avtalet där termen förekommer).

Termen `assume` är som `assert`, bara det att vid exekvering av termen, så sägs *subject* vara befriad från avtalet där assume-termen ingår om uttrycket *expression* inte håller. Om uttrycket håller, så måste *subject* genomföra de termer i avtalet vilka kommer efter assume-termen.

Utöver de fyra basala termtyperna, finns ytterligare ett antal termer som är inkluderade speciellt för tillämpningen i projektet. De presenteras härnäst.

`allocate` tar tre argument som säger vem allokerar, vem som är mottagare, och den konfiguration som allokeringen avser.

`delegate` tar tre termer som anger den som delegerar, vem som är delegaten och vilken insats det är som är delegerad.

`action` är namnet i prototypen på det vi i rapporten kallar insats. En action-term tar tre argument: vem som ska utföra insatsen, vilken insats som ska utföras, samt ett larm. Vid exekvering av en action-term triggas de avtal som styr formering av insatsgrupper och genomförande av insatser.

`ivar` (*internal variable*) är en term som fungerar som termen `assign` men den definierar och tilldelar endast variabler som är interna för det avtal i vilket termen förekommer.

`tell` och `hear` är två termer som sköter kommunikationen av värden på interna variabler mellan olika avtal. Termen `tell` tar fyra argument: vem det är som skickar information; ett namn på meddelandet, vad mottagaren ska göra med meddelandet, och själva meddelandet. I en `hear`-term finns två argument: namnet på den som lyssnar och namnet på meddelandet som denna ska lyssna efter. Två aktörer som ska kommunicera med varandra behöver alltså komma överens om ett namn på ett meddelande. En aktör som inte känner till namnet N på ett meddelande kan inte komma åt värdet på den interna variabeln som kommuniceras genom N.

Till sist har vi också definierat ett antal termer som styr exekveringen av avtalen:

`trig` tar ett argument vilket är ett villkor. Innebörden av `trig` är att exkveringen av ett avtal där en trig-term ingår stannar på trig-termen och inväntar att villkoret är uppfyllt. Tjänsteavtalen har t.ex. en trig-term som väntar på att ett larm utlöses.

`chop` är en term som instruerar avtalsexkveringsmekanismen att inte ta hänsyn till föregående terms duration. I nuvarande implementering av prototypen exekveras avtal i diskreta tidssteg: en term i varje tidssteg. `chop` möjliggör exekvering av flera på varandra följande termer i samma tidssteg).

`reset` är en term som återställer exekveringen av ett avtal till dess första term. Nedan ges ett exempel på en variant av `reset` där protokollet återställs till en godtycklig term. `reset` tillåter iterativ exekvering av protokoll.

Den sista termtypen vi tar upp är termer som i sig är protokoll. D.v.s. ett protokoll kan innehålla ett annat protokoll som en term.

Det finns två sätt att sätta samman termerna till protokoll:

- Seriellt, eller
- Parallellt

T.ex., vid körning av ett scenario sätts alla de definierade aktiveringsavtalen samman till ett parallellt protokoll. Det reflekterar själva idén med analysen: att analysera vilka resursaktiveringskonflikter som kan uppstå då flera resursaktiveringsplaners exekveras samtidigt. I prototypen är den huvudsakliga principen att aktiveringsplanerna är seriellt sammansatta.

Observera att vi har uteslutit förgrenade protokoll från de som kan konstrueras. Ett föregrenat protokoll skulle t.ex. kunna användas för att ge insatsansvariga möjlighet att välja en av flera alternativa åtgärder. Uteslutande

av förgrenade protokoll beror på två saker. För det första skulle det öka komplexiteten på algoritmen. För det andra skulle det innebära att prototypen skulle behöva implementera någon form av aktivt val hos de som utför protokollen, vilket förefaller vara både alltför avancerat för projektets omfång och inte nödvändigt för uppfyllande av projektets målsättning. Prototypen implementerar dock ändå en form av förgrenade protokoll och valmöjligheter genom att slumpvis välja ett av flera alternativ där så är möjligt. T.ex. vid val av delegat för en insats görs valet automatiskt på ett slumpartat sätt bland de aktörer som har ingått delegeringsavtal med en insatsansvarig.

Exemplet på ett avtal är hämtat direkt från Python-koden: det är en metod för att skapa ett tjänsteavtal (`svcctr`). Metoden skapar ett avtal där *offeror* erbjuder sig att utföra insatsen *act* inom tiden *dur* och med prioriteten *prio*, så fort ett meddelande med namnet *larmname* inkommer. Argumentet *name* är ett namn på tjänstekontraktet och *delegees* anger en lista (som kan vara tom) på aktörer som *offeror* vill delegera insatsen till.

```
def svcctr(self,name,offeror,larmname,act,dur,prio,*delegees):•

            p = terms.sequence(name=name)•        •
            p.trig(lam=0)•
            if delegees:
               p.dlg(offeror,act,*delegees)
            p.hear(offeror,larmname).resetpoint(1)•
            if delegees:
               p.act(offeror.delegate(act),act,larmname)•
            else:
               p.act(offeror,act,larmname)
            p.rst(1)
            self.addctr(p)
```

Vi går igenom metoden rad för rad. `p = terms.sequence(name=name)` skapar ett seriellt protokoll med namnet *name*. `p.trig(lam=0)` talar om att den första termen ska vara en trig-term och lam=0 anger att protokollet körs direkt när det blir anropat. (lam är en parameter som ifall den tilldelas ett icke negativt heltal gör att trig-termen väntar det antalet tidssteg i simuleringen med att exekveras. Ifall lam tilldelas ett positivt flyttal, så väntar trig-termen ett slumpvis antal tidssteg med att exekveras baserat på en exponentiell fördelning med lam som parameter. Om lam är ett villkor, så väntar trig-termen med sin exekvering tills villkoret är uppfyllt (vilket i sig aldrig behöver hända)).

```
            if delegees:
                p.dlg(offeror,act,*delegees)
```

Om det har angivits någon delegat, så adderas an delegeringsterm till protokollet `p` genom metoden `dlg`, som beskrivet ovan. I raden `p.hear(offeror,larmname).resetpoint(1)` adderas en term som säger att insatsansvarig (*offeror*) ska lyssna efter ett meddelande med namnet *larmname*. `resetpoint(1)` är en flödeskontrollterm som säger att när protokollet `p` återställs (vilket det gör i raden `p.rst(1)`) så påbörjas på nytt

exekveringen av protokollet p i termen precis innan den själv: allstå i det här fallet i `p.hear(offeror,larmname)`. Detta innebär att när *offeror* har satt igång en insats, så måste denne vara beredd på att sätta igång fler insatser om det kommer fler larm. Det därpå följande `if-else` uttryck lägger till en action-term till protokollet. Om det finns delegater, så blir det en av dessa som blir ansvarig för att utföra insatsen *`act`*. Exakt vem av delegaterna det blir, bestäms först under körningen av scenariot (run-time) eftersom det beror på flera dynamiska variabler och faktorer som inte kan kännas till då tjänsteavtalet definieras. Den sista raden i metoden ser till att protokollet p läggs till scenariot och kan fungera som underlag i en analys av resursaktiveringsproblem.

2.5 Algoritm

I det här avsnittet beskrivs algoritmen. Analysen består av två faser:

1. Genereringsfas
2. Konfliktanalys

I genereringsfasen simuleras en miljö där omgivningsvariabler (t.ex., i prototypscenariot, ifall det snöar eller inte och snönivån på de olika vägarna) uppdateras enligt givna och konfigurerbara regler. När värdena på omgivningsvariablerna blir oacceptabla (enligt konfigurerbara regler) utlöses ett eller flera larm vilka sätter igång insatsplaner och därmed också formeringsplaner. Dessa planer utgörs av exekverbara protokoll, vars exekvering simuleras. I protokollen finns termer som ger upphov till interaktioner mellan protokollen och de resurser som behövs för insatserna. Följande interaktionstyper är implementerade: *förfrågan*; *aktivering*; *avaktivering*; a*vslag*; *ingenAllokering*; och *aktiveringskonflikt*.

Interaktionerna mellan en resurs R och ett protokoll P ger upphov till så kallade *spår*. Ett spår som uppkommer av att protokollet P interagerar med resursen R identifieras av paret (R,P) och utgörs av en lista av tupler på formen (tid, interaktionstyp). (Detta är lite förenklat, tuplerna innehåller mer information, men för sammanhanget kan det utelämnas). Ett typiskt spår kan se ut som

```
(R,P): ( (8, förfrågan), (8, aktivering), (14, avaktivering) ).
```

Detta spår visar en lyckad resursaktivering. Resursen R förfrågades vid tidpunkten 8 via en term i protokollet P, blev direkt aktiverad, och återlämnades vid tidpunkten 14. Å andra sidan visar spåret

```
(R,Q): ( (9, förfrågan), (9, aktiveringskonflikt) )
```

på en resurskonflikt. Det finns alltså ett protokoll Q’ så att Q och Q’ är ger upphov till en aktiveringskonflikt. Med andra ord, fanns det i fallet med aktiveringskonflikt en allokering så att förfrågan i tidpunkten 9 via protokollet Q skulle ha resulterat i en frigörande av resursen R. Ägaren av R kunde här inte fullfölja sitt åtagande i allokeringsavtalet för att denne redan hade aktiverat samma resurs i samband med en förfrågan via ett annat protokoll Q’.

Spårets sammansättning (i.e. följd av interaktionstyper) ger en indikation på ifall ett resursaktiveringsproblem har uppstått, och i så fall också vilket problem det gäller.

Simuleringen av miljön och avtalsexekveringen körs över en bestämd tidsperiod (i prototypscenariot i 144 steg vilket motsvarar ett dygn uppdelat i 10-minutersperioder). För att få en rättvisande bild av resursaktiveringsproblemen i förekommande fall, körs samma scenario ett flertal gånger. Observera att miljön uppdateras dynamisk på grundval av slumpvariabler (så larmen kommer alltid vid olika tidpunkter från körning till körning) och även avtalsexekveringen är föremål för randomisering (i.e., med avseende på fördröjning av exekveringen och eventuellt andra indeterministiska val som kan göras i avtalens termer). På grund av detta blir spåren som generas i varje körning av den bestämda tidsperioden inte nödvändigtvis desamma som i föregående körningar.

Alla spår från alla körningar registreras i en databas.

I fas 2 analyseras de registrerade spåren. Detta görs genom att söka igenom alla spår efter de som innehåller någon interaktion som visar på problem i samband med resursaktivering. För varje sådant spår (R,Q), så genomsöks databasen igen efter de spår som interagerar med R samtidigt med Q. Resultatet är ett antal aggregat av spår som står i konflikt med varandra.

2.6 Problemets Komplexitet

Problemets komplexitet beror på antalet sätt på vilka resursaktiverings-planerna kan utföras. Det är antalet avtal (mer precist: antalet exekveringar av avtalen) som spelar den avgörande rollen, medan antalet termer i avtalen spelar en underordnad roll.

Vi ger en uppskattning av problemets komplexitet. Låt A vara en mängd av k stycken avtal med minst en term vardera, för ett $k > 1$. För att hitta alla möjliga konflikter behöver vi studera alla möjliga spår som kan uppstå i exekveringen av avtalen. Vi behöver alltså kunna analysera vad som händer för varje par (t,s) av termer från två olika avtal när de exekveras antingen i följden t:s eller i följden s:t (dvs den ena exekveras före den andra) eller när de exekveras samtidigt. Den minsta antalet tidpunkter sådant att alla kombinationer av termexekveringar kan representeras är då summan av alla termer i protokollen i A; kalla detta tal för M. Observera att M är begränsat underifrån av antalet avtal i A (varje avtal innehåller minst en term). Varje avtal P i A med n termer kan allstå utföras på minst M över n sätt (x över y är antalet sätt man kan välja x saker av y möjliga). M är strikt större än antalet termer i vart och ett avtalen i A, så M över n är minst lika stort som M, för varje n. Om vi antar att avtalen exekveras oberoende av varandra (som i det värsta fallet), då är antalet sätt som avtalen parallellt kan exekveras på begränsat underifrån av M^k. M är strikt större än k, så en undre begränsning av M^k är k^k, eller med andra ord: antalet avtal upphöjt i antalet avtal.

2.7 Prototypimplementering

Prototypen implementerar algoritmen beskriven ovan. Prototypen består av ett inmatningsspråk, en simuleringsmekanism, en analysdel, och en representation av resultatet av analysen. Dessutom finns det en given modell för några av beståndsdelarna i scenariot. Dessa delar beskrivs nedan.

Den givna modellen
Scenariot har tio vägar med fyra omvärldsvariabler:

1. *snödjup*, styrs av en förprogrammerad generator. Generatorn är en Markov-kedja som bestämmer när snön faller och när det är uppehåll. Den favoriserar snöfall. Om snön faller, så ökar snödjupet med en normalfördelning kring 3 cm i timmen (snödjupet ökar olika mycket i varje steg för de olika vägarna). Då scenariot startar initieras snödjupet på varje väg enligt en normalfördelning kring 3 cm.

2. *halka*, en binär variabel som styrs av en slumpprocess. Efter att en väg har blivit halkbekämpad tar det cirka 4 timmar innan halka uppstår igen.

3. *förekomst av snövallar*, en binär variabel. Då det har snöat 35 cm på en väg anses snövallar ha bildats.

4. *förekomst på väg av person i behov av medicinsk tillsyn* (hädanefter bara *skadad på väg*), binär variabel, styrs av en slumpprocess.

Vi tar för givet att det finns beskrivet ett antal insatser som kan utföras för att åtgärda problem med för mycket snö på en väg; halka på en väg; snövallar; och skadad på väg. I den beskrivningen antas också finnas angivet vilka resurser som behövs för att utföra dessa insatser. Se också avsnitt 2.1.3 för en beskrivning av insatsmodellen m.a.p. andra attribut på insatser.

Inmatning
Inmatningssätten och deras syntax är beskrivna i handboken för prototypen, vilket är en bilaga till rapporten. Vad som behöver matas in beskrivs i följande lista. Observera att inmatning sker genom inläsning av konfigurationsfiler, till vilket det finns interaktivt stöd och hjälp för i prototypen.

1. Ett antal konfigurationer vilka definierar de resursinstanser (både sammansatta och atomära) som finns tillgängliga i scenariot. T.ex. skapas av konfigurationen `config(driver, 10)` 10 stycken förare att tillgå i scenariot.

2. För de sammansatta resursinstanserna anges deras sammansättning, t.ex. säger uttrycket `plog=[config(driver, 1), config(fuel, 1)]` att en plog kan aktiveras endast om man också aktiverar en förare och en enhet bränsle. (Man kan göra dessa definitioner mer realistiska om man vill).

3. För varje konfiguration angiven i punkt 2 anges sedan hur många skilda leverantörer det ska finnas för denna konfiguration. Leverantörerna namnges automatiskt.

4. Från indata som har getts i punkterna 1-3 distribueras resursinstanserna mellan leverantörerna på ett slumpvis, men någorlunda rättvist, sätt. Vidare skapas ett antal allokeringskontrakt på ett slumpvis sätt mellan de som levererar sammansatta resurser och de som levererar komponenter till de sammansatta resurserna. Detta förfarande modellerar oberoendet i definierandet av krishanteringsorganisationernas resursaktiveringsplaner.

5. Om man så önskar, kan man ange en del av formeringsplanerna i detalj och på så vis kringgå de slumpvis definierade allokeringsavtalen. Det kan vara användbart om man känner till dessa planer och vill analysera dem med avseende på potentiella resurskonflikter.

6. Ett antal larmavtal som definierar en indikator baserat på en av de fyra omvärldsvariablerna (i.e. ett villkor för när ett larm ska utlösas).

7. Ett antal tjänsteavtal som

   - identifierar ett larm
   - pekar ut en insats och en insatsansvarig som ska svara på larmet och
   - anger de krav som ställs på tjänsten, t.ex. att insatsen ska vara avslutat inom en viss tid
   - samt anger till vilka den insatsansvariga har delegerat utförandet av insatsen.

8. För varje aktör som är delegerad en insats ska det anges en formeringsplan för hur denna ska sätta samman insatsgruppen som ska utföra insatsen. Formeringsplanen utgörs av ett antal allokeringsavtal.

9. Man kan om man vill som indata specialisera insatsmodellen för en given insats och en given utförare. T.ex. om en viss utförare är känd för att vara långsam kan modellen för hur lång tid insatsen kan ta justeras just för denne.

10. Övrig inmatning. Man kan sätta antalet tidssteg i scenariot (default 144). Man kan också sätta antalet körningar av scenariot (default 10).

Simuleringsmekanism

Prototypen körs genom simulering. När inmatningen är utförd, initieras scenariot, den första körningen (eller Run(0)) och en simuleringsmekanism startas för Run(0). En körning är en loop i 144 steg (eller det antal som angivits) där följande händer:

1. Den global klocka stegas upp (initialt sätts den till 0)

2. De lokala klockorna i varje protokoll stegas upp (initialt sätts de till 0)

3. För alla protokoll som har exekverbara termer, exekveras dessa termer parallellt (inte i parallellt i riktig mening, men sekventiellt så att effekten blir som parallellism). En term är exekverbar om den är den första termen i protokollet som inte har exekverats och inte är en trig- eller hear-term vars villkor ej är uppfyllt. Om protokollet har blivit brutet är ingen term i det exekverbar. Exekvering av en term kan resultera i Termination (term-exekveringen lyckades) eller Violation (term-exekveringen misslyckades) (Plus ett antal andra resultat som utelämnas här). Om term-exekveringen lyckas så ökas protokollets stegräknare (om det går, annars sägs hela protokollet vara terminerat). Om term-exekveringen misslyckas, så sägs protokollet vara brutet. Proceduren i det här steget med att alla termer som kan exekveras också exekveras är ett antagande som minskar komplexiteten av algoritmen. Alternativet skulle vara en indeterminism om när termerna exekveras.

4. De lokala klockorna i varje generator, infrastruktur och indikator stegas upp och omvärldsvariablerna uppdateras enligt den nya tiden, de effekter som protokollexekveringen har på dessa, och slumpprocesser som styr värden på omgivningen.

5. Det som hänt i protokollen, termerna och omgivningen registreras och en ny iteration med en ny körning påbörjas.

Analysdel
Analysen och genereringen av spår beskrevs i avsnittet 2.5.

Representation av resultat
Analysen och loggdata från det givna antalet körningar presenteras som text i en loggfil. För specialfallet då man gör endast en körning av scenariot finns det en visualisering av förloppet. I prototypen kan man ange om man vill se den eller inte. Den loggdata som kan fås fram är:

1. Hur omvärldsvariablerna förändrat värden under en körning

2. Status (tillgänglig eller inte) för varje resurs över tiden

3. Status för varje aktör över tiden

4. Insatsförlopp som beskriver vad som händer från att ett larm utlöses, till det att en insats är utförd för att svara på larmet. Här ses också vilka resursinstanser och aktörer som har varit inblandade i insatsen, liksom om insatsen utfördes enligt de uppställda kraven på t.ex. hur lång tid den skulle få ta.

5. Statistisk på resursutnyttjande.

Analysen presenteras som en lista av alla aggregat av avtal vilka någon gång under någon körning har gett upphov till en aktiveringskonflikt. För en körning kan det se ut som:

```
--------------------- run 5 ----------------------

1. Resource: DriverA, (rsi, 33)•(driver):
aktivation conflict of type flat out:
at time 78 in contract traces (ptr, 1102) and (ptr, 1162)

2. Resource: LoaderA, (rsi, 16) (loader):
aktivation conflict of type flat out:
at time 25 in contract traces (ptr, 1002) and (ptr, 1322)
at time 100 in contract traces (ptr, 1002) and (ptr, 1322)

--------------------- run 6 ----------------------
no conflicts
```

De upptäckta konflikterna summeras på slutet

```
--------------- conflict summary -----------------

number of runs: 10
time: 12.63 seconds

Resource (rsi, 33): 6 conflicts in 10 runs, 1 conflicting aggregates

Resource (rsi,16): 30 conflicts in 10 runs, 2 conflicting aggregates
```

2.8 Tolkning av resultat från körningar av prototypen

Kalkylen ger en sammanställning av de upptäckta konflikterna där den talar om vilka protokoll som har gett upphov till konflikterna och hur många gånger det har hänt. För att kunna tolka ett sådant resultat behöver vi veta vad det innebär att en konflikt om samma resurs uppkommer flera gånger mellan ett antal protokoll över flera körningar. Förekomsten av konflikt kan tolkas i termer av det som varieras mellan körningarna. I nuvarande implementering av kalkylen varieras tre parametrar slumpvis mellan körningarna:

1. Den tid det tar för en insats att genomföras (varierar enligt en normalfördelning)
2. Fördröjning av exekvering av insatsplaner (enligt en exponentiell fördelning).
3. Tiden för resursbehovet.

En tolkning av att en konflikt förekommer i alla körningar är att två avtal står i en allvarligare konlikt med varandra med avseende på tid för när en given resurs behövs. Eftersom vi har varierat både durationen på resursanvändningen och de tider som resursen efterfrågas, torde förekomsten av konflikt i varje körning tyda på att behoven är knutna till varandra på ett sådant sätt att resurskonflikt uppstår även om tiderna för resursbehoven och förfrågningarna ändras.

Att en konflikt förekommer i några körningar, men inte i alla, mellan två avtal om samma resurs torde tyda på att det kan finnas sätt att undvika konflikten genom att reglera de tider för vilka resursen får efterfrågas och användas för de två inblanda aktörer som exekverar de två avtalen i konflikt.

Att det inte förekommer någon konflikt mellan två avtal skulle kunna betyda att de två avtalen kan exekveras samtidigt utan att ge upphov till resurskonflikter och att det kan ske med viss marginal vad gäller tid för exekvering.

En viktig faktor i tolkningen av resultatet är troligen antalet körningar av samma scenario. Klart är att antalet körningar måste vara så pass stort att förhållandet mellan antalet upptäckta konflikter och antalet körningar tillåts konvergera till något värde med en viss felmarginal. Inget arbete har lagts ned i projektet för att studera dessa statistiska egenskaper hos kalkylen. Frågan om hur upptäckten av konflikter i resursaktivering ska tolkas är dock viktig, och det är troligen fel tolka resultaten på ett naivt sätt: om en konlikt är lättavhjälpt så kanske ett stort antal av sådana konflikter inte är så allvarlig, medan förekomsten av endast en svåravhjälpt konflikt kanske är den som fordrar uppmärksamhet.

I avsnitt 3 nedan är tolkning av resultat av analysen medtaget som en möjlig fortsatt forskning kring föreliggande ansats.

3. Inriktning på fortsatt forskning och utveckling

I detta avsnitt ges ett antal förslag på fortsatt forskning och utveckling av projektets ansats.

Tillämpning på specifik domän
Problemets komplexitet gör att det (antagligen) inte finns en generell lösning som kan tillämpas på alla möjliga domäner. Genom att studera en specifik domän, kan man söka en lösning som är gångbar just i den domänen. En inriktning på fortsatt forskning skulle alltså kunna vara att specialstudera en viss domän, och vidareutveckla kalkylen på ett sätt som är anpassat till de förhållanden som råder just där. Då kan också högre krav på realistiska parametriseringar av resurser och omgivning meningsfullt ställas.

Beaktande av fler källor till konflikter
Kalkylen kan hitta konflikter som beror på problem med allokering och tillgänglighet av resurser. Ett sätt att utvidga kalkylen skulle kunna vara implementera funktionalitet för att även hitta andra typer av problem i resursaktivering. Däribland innefattas

1. konflikter som härrör från de mekanismer som styr kopplingen mellan resurser och behov, t.ex. sådana som kan uppstå genom lokalt bestämda prioritering och resursdelningsprinciper

2. konflikter som kan uppstå på grund av avtalsbrott

Grunder för prioritering och resursdelning
Kalkylen kan ge ett underlag för hur resurser kan prioriteras och delas på ett samhällseffektivt sätt. Ett möjligt förfarande skulle kunna vara att iterativt förbättra prioriteringar eller delningsprinciper baserat på tester av sådana genom att använda den framtagna kalkylen. En nödvändig utvidgning av kalkylen skulle då vara att inkorporera mått på samhällsnytta och effektivitet.

Tolkning av resultat från körningar av kalkylen
En förfinad procedur för att registrera resursaktiveringsproblem (t.ex. genom att möjliggöra registrering av flera interaktionspunkter mellan avtalsexekvering och resurser tillsammans med en förfinad simuleringsmodell (t.ex. genom välja mer passande slumpfördelningar skulle kunna ge förbättrade möjligheter att tolka resultaten. Kalkylen skulle då kunna utvidgas och besvara frågor som:

1. om konflikter har upptäckts, kan de befintliga avtalen omformas eller kompletteras så att konflikter kring befintliga resurser inte uppstår?

2. om konflikter inte har upptäckts, hur robusta är aktiveringsplanerna i förhållande till en given kombination av behov, och befintliga resurser?

Referenser

1. Fredholm, L, Göransson, A (2006), *Ledning av räddningsinsatser i det komplexa samhället*, Räddningsverket 2006.
2. STIL-ramverk, En ansats för samverkan i samhället, Version 9-5, Försvarsmakten 2007
3. Back, R J R, Wright J von (2000), *Contracts Games and Refinement, Information and Computation*, 2000
4. Bjurling, B., *Contracts for Resources in Crisis Management*, Proceedings of the 7th International ISCRAM Conference – Seattle, USA, May 2010
5. Python, http://www.python.org/
6. SimPy, http://simpy.sourceforge.net/
7. Sadighi Firozabadi, B. *Decentralized Privilege Management for Access Control*. PhD Thesis, Department of Computing, Imperial College London, UK. November 2005.
8. Bandmann O, Dam M, Sadighi Firozabadi B., *Constrained Delegations*. In proceedings of IEEE Symposium on Security and Privacy, 2002.

Myndigheten för samhällsskydd och beredskap
651 81 Karlstad Tel 0771-240 240 www.msb.se