UNIVERSITY
OF TRENTO

DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY

38050 Povo – Trento (Italy), Via Sommarive 14
http://www.dit.unitn.it

Risk Analysis as part of the Requirements Engineering Process

Yudistira Asnar (yudis.asnar@dit.unitn.it)
Paolo Giorgini (paolo.giorgini@dit.unitn.it)

March 2007

Technical Report # DIT-07-014

Risk Analysis as part of the Requirements Engineering Process

Abstract

In software engineering, risk is usually considered and analyzed during, or even after, the system design. Countermeasures are elaborated and then accommodated as a refinement of the design, when a limited number of changes are still possible and they may introduce the problem of revisiting the initial requirements. In this paper, we propose a goal-oriented approach for modeling and reasoning about risk during the requirements analysis process. Risks are introduced and analyzed along the stakeholders' goals and countermeasures are introduced as part of the system's requirements. The approach extends the Tropos formal framework with new concepts and qualitative risk reasoning mechanisms. We use a case study on loan origination process to illustrate the proposal.

1 Introduction

Traditionally, in software engineering risk analysis is used to identify the high risk elements of a project and provide ways of documenting the impact of risk mitigation strategies [26]. Risk analysis has been also shown important in the software design phase to evaluate criticality of the system [4], where risks are analyzed and necessary countermeasures are introduced. Usually, countermeasures correspond to a design/system fine-tuning and then with a limited margin of change. However, it may happen that the risk reduction results in the revision of the entire design and possibly of the initial requirements, introducing thus extra costs for the project.

Considering risk since the early phases of the software development process can be useful to prevent such problems and, as effect, to contain costs [9]. Particularly, analyzing risk along stakeholders' needs and objectives, namely before requirements elicitation, can introduce good and valuable criteria to evaluate and choose among different alternative requirements.

Goal-oriented requirement engineering is an emerging research area where the concept of goal is used to model early requirements and non-functional requirements for a software system. The use of goals facilitates the analyst to understand the objectives of stakeholders and then motivate within the organizational setting the system's requirements. KAOS [10], *i** [36], GBRAM [2] and Tropos [5] are examples of goal-oriented methodologies and frameworks that have recently gained popularity in the community.

Particularly, Tropos is a requirement-driven development methodology based on the *i** modeling framework. Tropos proposes an early requirements analysis phase, where the analyst identifies the relevant stakeholders and models them as social actors, who depend on one another for goals to be fulfilled, tasks to be performed, and resources to be furnished. Through these dependencies, one can answer *why* questions, besides *what* and *how*, regarding system functionalities/requirements. Answers

to *why* questions ultimately link system functionalities to stakeholders' needs, preferences and objectives. Moreover, the methodology analyzes goals by a refinement process in which each goal is decomposed into subgoals and positive/negative contributions are established among goals. For example, in the case of a loan origination process the goal of assess loan applications can be OR-decomposed in assessed by in-house and assessed by Credit Bureau[1], while the goal receive electronic application may help (i.e., contributes positively) to verify loan application, the bank can indeed validate automatically the application with appropriated agencies.

Through goal models, the analyst can analyze alternative ways for the satisfaction of stakeholders' goals and choose among them on the base on specific criteria, like for example minimum-cost [28]. However, Tropos, as well as other goal-oriented approaches, does not consider risk within its requirements analysis process, and it may happen that the cheapest alternative corresponds to the most risky one. For instance, suppose that in order to receive loan application the bank can either receive electronic application or receive hard-copy application. Although, the electronic solution can economize the loan originating process, it can introduce a high level of risk due, for example, to the forgery of the application by external hackers or even by internal employees.

In this paper, we propose a goal-oriented approach for modeling and reasoning about risk at requirements level. The approach is based on Tropos methodology and proposes and extension of the original Tropos goal modeling and reasoning framework introducing a three layers model: goal, event, and treatment. Goals are AND/OR-decomposed and related to external events that can influence negatively (risk) their satisfaction. Treatments are then introduced to mitigate the effects of such events. We propose qualitative risk reasoning techniques to support the analyst in evaluating and choose among different possible sub-goals-trees.

The rest of the paper is organized as follows. In Section 2, we present the related work and our main contribution. Section 3 introduces the *Loan Origination Process* case study that is used to describe the goal-risk analysis framework (Section 4). The risk analysis process and algorithms for qualitative reasoning are presented in Section 5, while in Section 6 we describe the developed CASE tool along some experimental results. Finally, we conclude the paper with a final discussion in Section 7.

2 Background

2.1 Related Work

Related work lies on three major areas: requirement engineering, secure and dependable engineering, and risk analysis.

In **requirement engineering**, Dardenne et al. [10] propose KAOS, a goal-oriented requirements engineering methodology aiming at modeling not only *what* and *how* aspect of requirements but also *why*, *who*, and *when*. KAOS introduces also the concept of *obstacles* [35] and *anti-goal* [34] which can be seen as boundaries in requirement analysis. An obstacle is defined as an undesirable behavior to strategic interests of stakeholders, and an anti-goal defines a goal that belongs to an attacker that obstructs the fulfillment of stakeholders' goals. In other word, obstacles can be seen as unintentional-risk, since risk is an undesirable behavior, and anti-goals are threats or intentional risks. These features make KAOS suitable for analyzing requirements of

[1]The company that provides credit information and assesses loan applications

secure and dependable system. In [35], van Lamswerdee and Latier demonstrate how to derive obstacles and guarantee their completeness from a goal structure and the analyst's knowledge about the system.

Mayer et al. [23] extend the *i** conceptual framework [36] to analyze risk and security issues during the development process of IT systems, requirement analysis in particular. The framework models the business assets (i.e., goals) of an organization and assets of its IT system (i.e., architecture, design decisions). Countermeasures to mitigate risks are then selected in such a way that the risks do not affect these assets. Liu et al. [20] propose a methodological framework for security requirements analysis based on *i**. They use the NFR framework [8] to support the formal analysis of threats, vulnerabilities, and countermeasures.

In the area of **secure and dependable system**, the most used frameworks are the classical ones, namely Fault Tree Analysis (FTA) [32], Failure Modes, Effects, and Criticality Analysis (FMECA) [1]. In security engineering, approaches like *attack tree* and *threat tree* [15, 27] are similar to the FTA, while others proposals like UMLSec [17], SecureUML [21], Abuse Case [24], and Misuse Case [31] are funded on UML as modeling language. However, the most relevant work for our purpose is Defect Detection and Prevention (DDP) by Feather et al. [12] has been developed and applied in Jet Propulsion Lab of NASA. DDP consists of a three layers model: Objectives, Risks, and Mitigation. Each objective has a *weight* to represent its importance, each risk has a *likelihood* of occurrence, while mitigation has a *cost* for its accomplishment (mainly resource consumption). Severity of a risk can be represented by an impact relation between the objective and the risk. Moreover, a DDP model specifies how to compute the level of objectives achievement and the cost of mitigations. This calculation allows one to evaluate the impact of taken countermeasures and then support the decision making process. Finally, a DDP model can be integrated with other quantitative frameworks (e.g., FMECA, FTA) in order to model and assess risks/failures [12].

Lee et al. [19] propose a framework for modeling critical systems (particularly, socio-technical systems) which is based on a standard developed by US Department of Defense (US-DoD), called DoD Information Technology Security Certification and Accreditation Process (DITSCAP) [33]. The DITSCAP framework assesses the risk caused by vulnerabilities and threats of a system by evaluating the implementation of security requirements.

In the area of **risk analysis**, uncertain events (i.e., threats and failures) are quantified with two attributes: likelihood and severity. Probabilistic Risk Analysis (PRA) [3] is widely used for quantitatively risk assessment, while approaches like FMECA [1] quantify risk into qualitative values: frequent, reasonable probable, occasional, remote, and extremely unlikely. Basically, events are prioritized using the notion of "expectancy loss" which is a multiplication between the likelihood of events and its severity. This priority represents the criticality of an event. When resources are limited, an analyst can decided to adopt countermeasures for mitigating events on the basis of their priority. However, identification of probabilities is not necessarily precise, and typically it strongly depends on expert judgments. Approaches like Multi-Attribute Risk Assessment [30] can improve the risk analysis process by considering multi-attributes. Many factors like reliable, available, safety and confidentiality can result critical for a system and each of them has its own risk value. This introduces the need for the analyst to find the right trade-off among these factors. For instance, an Air Traffic Management system is required to be always available and safe. Certain conditions (e.g., radar noise) can affect the normal behavior of the system and consequently

its safety. In many cases, the best solution is to restart the system. This, however, reduces the availability of the system. In [6], Butler presents how to choose cost-effective countermeasures to deal with existing security threats by using multi-attribute risk assessment.

Finally, CORAS [11] is aiming at developing a framework for risk analysis of security critical systems. The CORAS risk management consists of the following steps: context identification, risk identification, risk analysis, risk evaluation, and risk treatment.

2.2 Paper Contribution

Though, the literature offers a variety of contributions in the area of risk analysis, a lot must still be done to fully integrate these approaches in the software development process. Our aim in this paper is to propose a framework to analyze risk since the initial phases of the software analysis, particularly when the analyst needs to analyze not only the system-to-be but also the organizational setting in which it will operate. We strongly believe that an accurate risk analysis in this phase can improve the quality of the entire software project. Moreover, most of the frameworks presented in literature are mainly modeling frameworks without any, or a limited, capacity of reasoning and automated analysis. In this paper, we extend the Tropos goal model [13, 28] with concepts of risk and mitigation and we propose qualitative reasoning mechanisms to consider risk as an evaluation criterion during the requirement analysis process.

3 Case Study

The case study we use in this paper originated within the European project SERENITY[2] and focuses on a typical *Loan Origination Process* (LOP) that starts with the receiving a loan application and ends, possibly, with the loan approval. When the bank receives the application, it starts the process of verifying the data and calculating the credit rating. The rating can be obtained either by an internal rating (in-house assessment) or external rating (Credit Bureau). After the calculation, the bank defines the loan schema, namely it defines the loan cap and its interest. We assume that the loan schema is initially proposed by the customer, but is the bank that takes the final decision. The bank is, of course, also interested in ensuring the repayment of the loan and having more income.

Several uncertain events (i.e., threats and un/intentional events) can affect the success of the whole process. For example, forgery of the loan application, fake the identification document, ignorance of Credit Bureau about the bank condition, etc. Some of these events are considered unacceptable and must be avoided. To do this, it is needed to introduce some additional measures aiming at reducing the likelihood or the effects of these events. However, these measures imply extra costs for the whole process and they should be analyzed carefully before their adoption.

4 Tropos Goal Risk Framework

Tropos is a software development methodology that adopts the concept of agent and its related mentalistic notions (e.g., goals, tasks, and resources) along all the phases of

[2] http://www.serenity-project.org/

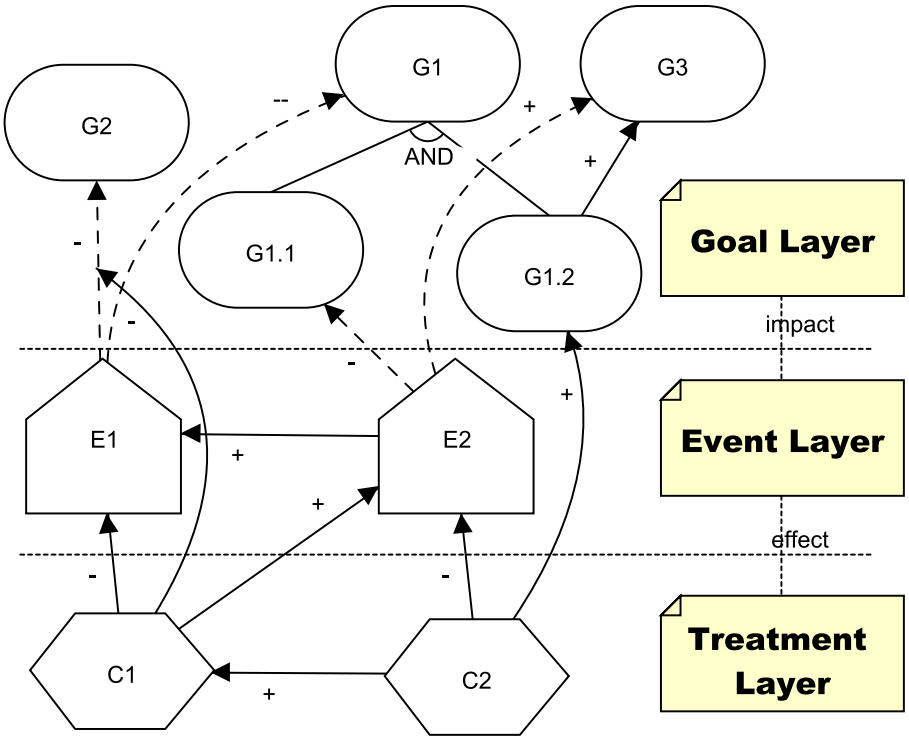

Figure 1: Goal-Risk Model

the development process [5]. The methodology spans from early requirements analysis up to implementation and uses goal models to represents agent (or more general actor) mental states [14]. The key role of early requirements analysis is to model the system-to-be together with the organizational setting where the system operates. In the following we extend the Tropos goal modeling framework [13, 28] by introducing constructs and relations specific for analyzing risk.

A Goal-Risk (GR) model is represented as a graph $\langle \mathcal{N}, \mathcal{R} \rangle$, where $\mathcal{N}$ are nodes and $\mathcal{R}$ are relations (see Fig. 1). $\mathcal{N}$ is comprised of *goals*, *plans*, and *events*. Goals (depicted as ovals) are strategic interests that actors intend to achieve. Events (depicted as pentagons) are uncertain circumstance which is out of the control of actors that can have an impact (positively or negatively) on the fulfillment of goals. Tasks (depicted as hexagons) are sequences of actions used to achieve goals or to treat the events[3].

Each constructs has two attributes: SAT- $Sat(G)$ and DEN- $Den(G)$. Such attributes represent respectively the value of evidence that the construct will be satisfied or denied. In the probability theory, if $Prob(A) = 0.1$ than we can infer that probability of $\neg A$ is 0.9 (i.e., $P(\neg A) = 1 - P(A)$). Conversely, based on the idea of Dempster-Shafer theory [29] the evidence of the goal being denied (DEN) can not be inferred from the satisfaction evidence (SAT) and vice versa. For instance, the bank has the goal verify loan application, and the goal is affected by the event of having fake

[3]In this paper, we use hexagon **only** to denote an event treatment, and **not** to define as means to achieve a goal.

identity document. The only conclusion, we can infer, is the goal has DEN which is introduced from the event, while we can not say anything about SAT because there is no piece of information that can be categorized as satisfaction evidence. These attribute values are qualitatively represented in the range of *(F)ull, (P)artial, (N)one*, with the intended meaning $F > P > N$. *Full* (*Partial, None*) evidence for the satisfaction of a gaol means that there is (at least) "sufficient" ("some", "no") evidence to support the goal to be fulfilled. Analogously, *Full* (*Partial, None*) evidence for the denial of a goal means that there is (at least) "sufficient" ("some", "no") evidence to support the goal to be denial.

Relations $\mathcal{R}$ are represented as $(N_1, \ldots, N_n) \stackrel{r}{\longmapsto} N$, where r is the type of the relation, $N_1, \ldots, N_n$ are called *source nodes* and N is the *target node*. r consists of *AND/OR-decomposition*, *contribution*, *alleviation*, and *impact* relations. AND/OR decomposition relations are used to refine goals, tasks, and events in order to produce a finer structure. Contribution relations are used to model the impacts of a node over another node. Basically, Tropos distinguishes 4 types of contribution relations: $+$,$++$,$-$, and $--$. Each type can propagate either evidence for SAT or DEN or both. For instance, the "$++$" contribution relation indicates that the relation propagates both SAT and DEN evidence, and the "$++_S$" contribution relation means the relation only propagates SAT evidence towards target nodes. The same intuition is applied for the other types of contribution in delivering DEN evidence. However, the "$--$" and "$-$" propagate the crossing evidence. For instance, "$-$" propagates the SAT value of the source node to the DEN value of the target node, and vice versa. Alleviation relations are used to model the severity reduction of events. Essentially, alleviation relations are the same of contribution relations (i.e., $+$,$++$,$-$,$--$), but with a different semantic (see later for the formal definition). Finally, an *impact* relation (depicted as dash line-arrow) represents the severity of an uncertain event in affecting the goal layer. This relation is categorized as part of $\mathcal{N}$ and $\mathcal{R}$ in $\langle \mathcal{N}, \mathcal{R} \rangle$. The detail of this relation and the reason why it is categorized as a vertex instead of only an edge is detailed in the next event layer subsection.

In the following subsections, we describe the three layers of the GR model through the loan origination process case study.

4.1 Goal Layer

The goal layer is adopted from Tropos goal model [13] which analyzes strategic interests of the stakeholders. As shown in Fig. 2, the modeling starts with identifying top goals of stakeholders (e.g., bank management) which are earn more income *(G_1)*, receive loan application *(G_4)*, ensure repayment of loan *(G_7)*, and handle loan application *(G_{10})*.

Each top goal is refined using AND/OR decomposition into subgoals. For example, G_1 is OR-decomposed into earn from loan interest *(G_2)* or charge high fee for loan origination *(G_3)*. This can be seen as a way to model an alternative, indeed to achieve G_1 , one can either fulfill G_2 or G_3. In Tab. 1 the rules used to propagate evidence trough AND/OR relations are presented (first two rows of the table). Thus, $Sat(G_1)$ is calculated on the base of maximum value among SAT values of all its subgoals (e.g., G_2 and G_3). Conversely, $Den(G_1)$ is defined as the minimum value between DEN values of its subgoals. However, G_{10} is refined (AND) into verify loan application *(G_{11})*, assess application *(G_{12})*, define loan schema *(G_{15})*, and approve loan application *(G_{18})*. It means that to achieve G_{10} , all its subgoals (i.e., G_{11}, G_{12}, G_{15}, and G_{18}) must be satisfied. This decomposition process continues until

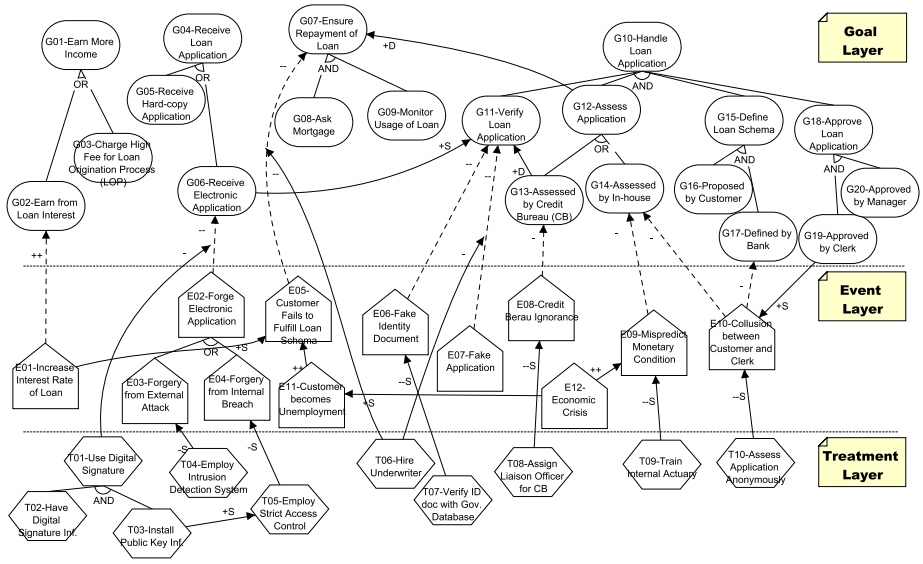

Figure 2: Goal-Risk Model for Loan Originating Process

all leaf goals are tangible (i.e., there is an actor that can fulfill it).

The next step is to model interrelations among goals using contribution relations. For instance, the goal receive electronic application *(G_6)* supports the goal verify loan application *(G_{11})* (i.e., $G_6 \overset{+S}{\longmapsto} G_{11}$) because it promotes the possibility of doing automatic verification. As presented in Tab. 1, "$+$" and "$-$" relation propagates at most *Partial* evidence from source nodes, while "$++$" and "$--$" may propagates at most *Full* evidence. Differently, from [13] in GR model the goal layer may effect other layers. This allows us to model situations where a goal fulfillment increases/reduces the occurrence of an event or a goal fulfillment supports/prevents countermeasures accomplishment. These two inter-layer relations are modeled using contribution relations.

4.2 Event Layer

In the GR framework we adopt the WordNet[4] definition for event:

- something that happens at a given place and time;
- a special set of circumstances;
- a phenomenon located at a single point in space-time;
- a consequence; i.e., a phenomenon that follows and is caused by some previous phenomena.

Essentially, the notion of event, here, is slightly different from *threat* [25] in computer security and *hazardous* condition in reliability engineering [22]. Those concepts are only defined as a potential circumstance that could cause harm or loss and not specifying the notion of likelihood.

[4]http://wordnet.princeton.edu/

Relation	$Sat(N_1)$	$Den(N_1)$
$(N_2, N_3) \overset{and}{\longmapsto} N_1$	$min \left\{ \begin{array}{l} Sat(N_2), \\ Sat(N_3) \end{array} \right\}$	$max \left\{ \begin{array}{l} Den(N_2), \\ Den(N_3) \end{array} \right\}$
$(N_2, N_3) \overset{or}{\longmapsto} N_1$	$max \left\{ \begin{array}{l} Sat(N_2), \\ Sat(N_3) \end{array} \right\}$	$min \left\{ \begin{array}{l} Den(N_2), \\ Den(N_3) \end{array} \right\}$
$N_2 \overset{+_S}{\longmapsto} N_1$	$min \left\{ \begin{array}{l} Sat(N_2), \\ P \end{array} \right\}$	N
$N_2 \overset{++_S}{\longmapsto} N_1$	$Sat(N_2)$	N
$N_2 \overset{+_D}{\longmapsto} N_1$	N	$min \left\{ \begin{array}{l} Den(N_2), \\ P \end{array} \right\}$
$N_2 \overset{++_D}{\longmapsto} N_1$	N	$Den(N_2)$
$N_2 \overset{-_S}{\longmapsto} N_1$	N	$min \left\{ \begin{array}{l} Sat(N_2), \\ P \end{array} \right\}$
$N_2 \overset{--_S}{\longmapsto} N_1$	N	$Sat(N_2)$
$N_2 \overset{-_D}{\longmapsto} N_1$	$min \left\{ \begin{array}{l} Den(N_2), \\ P \end{array} \right\}$	N
$N_2 \overset{--_D}{\longmapsto} N_1$	$Den(N_2)$	N

Table 1: Evidence Propagation Rules for (AND/OR) Decomposition and Contribution Relations

Following the Probabilistic Risk Assessment (PRA) approach [3], the GR framework characterizes events with two properties: likelihood and severity. Likelihood is modeled as a property of an event, whereas severity is denoted as the sign (negative/positive) of an impact relation. This representation allows us to model situations where an event impacts on more than a single goal. For instance, in Fig. 2 the event collusion between customer and clerk *(E_{10})* obstructs the satisfaction of the goal defined by bank *(G_{17})* in defining loan schema because the officer may not be objective. On the other hand, it also obstructs the goal assessed by in-house *(G_{14})* since it can compromise the integrity of the employees that assess the loan application. Moreover, an event becomes a risk when it produces a negative effect, alternatively an opportunity when it produces positive effects. This flexibility allows an analyst to model an event which acts as a risk and an opportunity at the same time. For instance, in Fig. 2 the event increase interest rate of loan *(E_1)* can be seen as a risk for the goal ensure repayment of loan *(G_7)* and as an opportunity for the goal earn from loan interest *(G_2)*. The analyst should realize that it is not convenient to eliminate totally the risk, since the event introduces also advantages, but rather it is better to mitigate its negative effects.

The identification of the events can be realized using different approaches, such as obstacle analysis [35], anti-goal [34], hazard analysis [18], misuse case [31], abuse case [24], taxonomy-base risk identification [7], or risk in finance [16]. Afterwards, each event is decomposed into sub-events until each leaf event can be easily assessed and paying attention to ensure that sub-events are disjoint events. To model dependency among events, one can use contribution relations, such as $E_{12} \overset{+_S}{\longmapsto} E_{11}$ in the case study.

As we have already said an event can be characterized by two properties: *likelihood* and *severity*. In our framework, we calculate the likelihood of an event ($\lambda(E)$) from the level of evidence that supports (SAT) and prevents (DEN) the occurrence of the event. The likelihood is represented qualitatively with the following values: *(L)ikely, (O)ccasional, (R)are*, and *(U)nlikely*, with intended meaning $L > O > R > U$. Tab. 2 defines the calculation rules of likelihood from SAT and DEN values. An event

$Sat(E) \wedge Den(E) \longmapsto \lambda(E)$		
$Sat(E)$	$Den(E)$	$\lambda(E)$
F	N	L
F	P	O
P	N	O
F	F	R
P	F	R
P	P	R
N	F/P/N	U

Table 2: Likelihood Calculation based on Evidence Values

$\lambda(E)$ / Impact Relation	L	O	R	U
	$Sat(G)$			
$E \overset{++}{\longmapsto} G$	F	P	P	N
$E \overset{+}{\longmapsto} G$	P	P	N	N
	$Den(G)$			
$E \overset{--}{\longmapsto} G$	F	P	P	N
$E \overset{-}{\longmapsto} G$	P	P	N	N

Table 3: Evidence Propagation Rules for Impact Relation

with *full* evidence of being satisfied and no evidence of denial implies a *likely* event. Consequently, an event without any evidence of satisfaction results to be an *unlikely* event, no matter the value of denial evidence.

By severity, we mean the influence of an event to the goal fulfillment. This definition is similar with the one in FMECA [1] or *impact* given in DDP [12]. This property is classified as follows:

- Strong Positive($++$) - the event occurrence produces a *strong* contribution to the goal satisfaction;
- Positive($+$) - the event occurrence produces a *fair* contribution to the goal satisfaction;
- Negative($-$) - the event occurrence produces a *fair* contribution to the goal denial;
- Strong Negative($--$) - the event occurrence produces a *strong* contribution to the goal denial.

This classification is encoded as the sign of an impact relation which connect the event layer with the goal layer. Impact relations introduce new evidence for the goal layer, and the value of new evidence depends on the likelihood of the event and the sign of impact relation (Tab. 3). The rule specifies that an event propagates *full* evidence of satisfaction to the goal layer if its likelihood is *likely* and connected by "$++$" impact relation (called an opportunity). Conversely, an event produces *partial* evidence of denial, when connected by "$--$" and has *occasional/rare* likelihood, or in the case it is connected by "$-$" with *likely/occasional* likelihood (called risk).

Impact relation is a hybrid construct in $\langle \mathcal{N}, \mathcal{R} \rangle$. In other word, it is categorized as a relation in $\mathcal{R}$, but it is also a node in $\mathcal{N}$. Analogously, we introduce another relation, called *alleviation*, that connects a treatment with the impact relation. This allows us to model a treatments as a mitigation for the reduction of the severity (e.g., $-- \longmapsto -$).

Alleviation \ Impact	Initial	Rewrite
$T \overset{=}{\longmapsto} [Impact]$	$E \overset{=}{\longmapsto} G$	$E \overset{\emptyset}{\longmapsto} G$
	$E \overset{-}{\longmapsto} G$	$E \overset{\emptyset}{\longmapsto} G$
$T \overset{-}{\longmapsto} [Impact]$	$E \overset{=}{\longmapsto} G$	$E \overset{-}{\longmapsto} G$
	$E \overset{-}{\longmapsto} G$	$E \overset{\emptyset}{\longmapsto} G$

Table 4: Rewriting Rules for Alleviation Relation

We can treat the impact relation as an end node of the alleviation relation. The detail about alleviation relations will explain in the next subsection.

4.3 Treatment Layer

Once the goal and event layers have been analyzed, the analyst identifies and analyzes the countermeasures to be adopted in order to mitigate risks in the GR model. Treatments/countermeasures can be analyzed using (AND/OR) decomposition and contribution relations. Essentially, the mitigation operates in two different ways: reducing the *likelihood* or reducing the *severity*. To reduce the likelihood, a countermeasure is modeled using a contribution relation which introduces denial evidence to the event. For instance, the treatment employ intrusion detection system *(T_4)* adds denial evidence for the risk forgery from external attack *(E_3)*, and consequently applying rules of Tab. 2 it results a less likely event.

To reduce the impact, we introduce the *alleviation* relation, as mention before. This relation intends to reduce the severity of the *impact* sign; in Fig. 2 for example, the relation between the treatment use digital signature *(T_1)* to the impact relation between the event Forge Electronic Application *(E_2)* and goal receive electronic application *(G_6)*. This relation is not intended to reduce the likelihood, but rather to reduce the severity of the risk E_2 (i.e., $T_1 \overset{-}{\longmapsto} [E_2 \overset{=}{\longmapsto} G_6]$) by rewriting $E_2 \overset{=}{\longmapsto} G_6$ into $E_2 \overset{-}{\longmapsto} G_6$. The production rules for this relation are presented in Tab. 4. For the sake of simplicity, we just consider whether the treatment is selected or not and we do not take the treatment evidence into account. Alleviation relations reduce only the "negative" impact relations, and $\emptyset$ indicates there is no impact between the event and the goal.

In our model, we also allow for relations between the treatment layer and the goal layer. This is useful to model situations where a countermeasure adopted to mitigate a risk has also a contribution (especially negative) to some goal. For instance, the countermeasure use digital signature can mitigate the event forge of electronic application, but it also introduces new additional costs that can be seen as denial evidence for the goal have low cost loan origination process.

5 Risk Analysis Process

In this section, we describe the methodological process and qualitative risk reasoning techniques used to analyze and evaluate alternative goal models. Particularly, the problem we focus on is to find and evaluate all possible ways to satisfy top goals with an acceptable level of risk. In other words, given a GR model, each OR decomposition introduces alternative modalities for top goals satisfaction, namely different sets of leafs

goals that can satisfy top goals. Each of these alternative solutions may have a different cost and may introduce a different level of risk. Risk can be mitigated with appropriate countermeasures, which, however introduce extra costs that have to be added to the cost of the solution.

The analysis process is described in the Algorithm 1 and consists of the following three steps:

1. find alternative solutions (line 2-3),
2. evaluate each alternative against relevant risks (line 5-6)
3. assess the countermeasures to mitigate risks (line 9-15).

The process starts taking in input the GR model, a set of desired satisfaction values for top goals ($desired\_labels$), acceptable risk values (acc_risks), and a number of goals as possible candidates for the final solution ($input\_goals$). For instance, we might desire to have *full* evidence for satisfaction of goals G_4 and G_{10} (i.e., $Sat(G_4) = F$ and $Sat(G_{10}) = F$), while we do not care about goals G_1 and G_7 admitting *partial* evidence for their satisfaction (i.e., $Sat(G_1) = P$ and $Sat(G_7) = P$). We might be also interested to not run any risk for G_1 and G_7 (i.e., $Den(G_1) = N$ and $Den(G_7) = N$) while we can admit *partial* evidence for the denial of G_4 and G_{10} (i.e., $Den(G_4) = P$ and $Den(G_{10}) = P$). As $input\_goals$, we may want to consider the set $\{G_2, G_3, G_5, G_6, G_8, G_9, G_{11}, G_{13}, G_{14}, G_{16}, G_{17}, G_{19}, G_{20}\}$.

Algorithm 1 Risk_Analysis_Process

Require: goal_model $\langle \mathcal{N}, \mathcal{R} \rangle$, label_array top_goals, node_array input_goals, label_array events
1: solution_array solution {solution that has already encompassed risks and necessary countermeasures}
2: alt_solution $\leftarrow Backward\_Reasoning(\langle \mathcal{N}, \mathcal{R} \rangle, desired\_labels,$ $\mathbf{nil}, input\_goals)$
3: candidate_solution $\leftarrow Select\_Can\_Solution(alt\_solution)$ {candidate_solution $\subseteq$ alt_solution}
4: **for all** $S_i \in candidate\_solution$ **do**
5: **if** $Satisfy(\langle \mathcal{N}, \mathcal{R} \rangle, desired\_labels, acc\_risks,$ $\langle S_i, events, \mathbf{nil} \rangle)$ **then**
6: $add(solution, \langle S_i, \mathbf{nil}, Calc\_Cost(S_i, \mathbf{nil}) \rangle)$
7: **else**
8: max_labels $\leftarrow Backward\_Reasoning(\langle \mathcal{N}, \mathcal{R} \rangle,$ $desired\_labels, acc\_risks, goals(S_i))$
9: cur_labels $\leftarrow Forward\_Reasoning(\langle \mathcal{N}, \mathcal{R} \rangle,$ $\langle S_i, events, \mathbf{nil} \rangle)$
10: rel_treatments $\leftarrow Find\_Treatments(\langle \mathcal{N}, \mathcal{R} \rangle,$ $cur\_labels, max\_labels)$
11: **for all** $C_j \in 2^{rel\_treatment}$ **do**
12: **if** $Satisfy(\langle \mathcal{N}, \mathcal{R} \rangle, desired\_labels, acc\_risks,$ $\langle S_i, events, C_j \rangle)$ **then**
13: $add(solution, \langle S_i, C_j, Calc\_Cost(S_i, C_j) \rangle)$
14: **end if**
15: **end for**
16: **end if**
17: **end for**

$Backward\_Reasoning$ (line 2) generates a set of possible assignment values of evidence for the input goals that satisfy the desired values ($desired\_labels$). Essentially, $Backward\_Reasoning$ is the top-down reasoning mechanism proposed in [28], where goal models are encoded into satisfiability formulas and then SAT solvers are used to find which input-goals can satisfy top goals. The use of the standard Tropos

backward reasoning is limited to the goal layer (i.e., not considering the relations with the other two layers), and without specifying any constraints. So for instance, in order to achieve $desired\_labels$ in our case study we have 8 alternative solutions (see Tab. 5) as follows:

- S1=$G_2, G_5, G_8, G_9, G_{11}, G_{13}, G_{16}, G_{17}, G_{19}, G_{20}$
- S2=$G_3, G_5, G_8, G_9, G_{11}, G_{13}, G_{16}, G_{17}, G_{19}, G_{20}$
- S3=$G_2, G_6, G_8, G_9, G_{11}, G_{13}, G_{16}, G_{17}, G_{19}, G_{20}$
- S4=$G_3, G_6, G_8, G_9, G_{11}, G_{13}, G_{16}, G_{17}, G_{19}, G_{20}$
- S5=$G_2, G_5, G_8, G_9, G_{11}, G_{14}, G_{16}, G_{17}, G_{19}, G_{20}$
- S6=$G_3, G_5, G_8, G_9, G_{11}, G_{14}, G_{16}, G_{17}, G_{19}, G_{20}$
- S7=$G_2, G_6, G_8, G_9, G_{11}, G_{14}, G_{16}, G_{17}, G_{19}, G_{20}$
- S8=$G_3, G_6, G_8, G_9, G_{11}, G_{14}, G_{16}, G_{17}, G_{19}, G_{20}$

with *full* SAT evidence for all goals except for G_2 , G_3 , G_8 , and G_9 which might be *partial* SAT. Among these solutions, the analyst chooses one of them $candidate\_solution$ (line 3) on the basis of a certain criterion, like for example minimum-cost [28]. Suppose, the analyst decides to choose alternatives with a cost less than 30 (S4, S7, and S8 in our case – see Tab. 5).

Each $candidate\_solution$ is now evaluated against risk and possibly necessary countermeasures are introduced (line 4-17). First, the analyst checks whether the $candidate\_solution$ (e.g., S4), together with risks in the event layer ($events$), still can obtain the desired values of evidence for top goals. To do this, $Satisfy$ uses the $Forward\_Reasoning$ mechanism adapted from [13] which propagates evidence values of inputs throughout the goal model. Final evidence values for top goals are compared with those initially desired. If DEN values for top goals are equal/less than the maximum risk admitted (acc_risks) and SAT values for top goals are equal/great than the desired values ($desired\_labels$), then the $candidate\_solution$ is added directly to the $solution$ and its cost is calculated (line 6). Otherwise, countermeasures must be introduced in the $candidate\_solution$ (line 10-15). The adaptation of $Forward\_Reasoning$ will detail at the end of this section.

In order to define countermeasures, the analyst calculates the maximum DEN values of input goals (max_labels) that produce acceptable DEN values for top goals. This means we need to find a set of countermeasures able to mitigate risk, so that we have at most the acceptable risk level (acc_risks). We calculate max_labels values using $Backward\_Reasoning$ (line 8) and specifying the goals of the $candidate\_solution$ as input goal and acc_risks as a constraint for the DEN value of input goals. $Forward\_Reasoning$ (line 9) propagates evidence values of the $candidate\_solution$ and $events$ throughout the model, so that the evidence values of all goals are defined (cur_labels). By having two set of values (i.e., cur_labels and max_labels), we can find treatments to mitigate risks.

To this end, we propose $Find\_Treatments$ (Algorithm 2) which enumerates all the possible set of treatments that may mitigate risks. By comparing DEN in the max_labels and the cur_labels, one can identify which goals are overvalued. For instance, given S4 (Tab. 5) the goal G_6 in max_labels is defined as $Den(G_6) = P$, whereas in cur_labels is $Den(G_6) = F$ (see Tab. 6 column Event-Out). Afterwards,

Algorithm 2 Find_Treatments

Require: goal_model $\langle \mathcal{N}, \mathcal{R} \rangle$, label_array current, label_array max
1: array rel_events, rel_alleviations, rel_treatments
2: **for all** $c_i \in current \wedge isGoal(c_i)$ **do**
3: **if** $c_i.den > max[i].den$ **then**
4: tmp_events $\leftarrow related(\langle \mathcal{N}, \mathcal{R} \rangle, c_i)$ {find all related events of goal c_i}
5: $add(rel\_events, tmp\_events)$
6: **for** $e_j \in tmp\_events$ **do**
7: **for** $R_k \in \mathcal{R}$ s.t. $source(R_k) = e_j \wedge target(R_k) = c_i$ **do**
8: tmp_alleviations $\leftarrow related(\langle \mathcal{N}, \mathcal{R} \rangle, R_k)$ {find all relevant allevation relations of goal c_i}
9: $add(rel\_alleviation, tmp\_alleviation)$
10: **end for**
11: **end for**
12: **end if**
13: **end for**
14: **for all** $e_i \in rel\_events$ **do** {find all possible treatments for reducing likelihood of related events}
15: tmp $\leftarrow related(\langle \mathcal{N}, \mathcal{R} \rangle, e_i)$
16: $add(rel\_treatments, tmp)$
17: **end for**
18: **for all** $a_i \in rel\_alleviation$ **do** {find all possible treatments for reducing severity of related events}
19: **for all** $R_j \in \langle \mathcal{N}, \mathcal{R} \rangle$ s.t. $R_j = a_i$ **do**
20: $add(rel\_treatments, [source(R_j)])$
21: **end for**
22: **end for**
23: **return** treatments

we need to define the related events that may cause this level of risk[5]. In our case, the relevant events for G_6 are E_2 , E_3 , and E_4 . Once we have identified the rel_events, we can find the treatments ($rel\_treatments$) that can mitigate these events (line 14-22). As we discuss before, the mitigation operates reducing likelihood and severity. In the case of E_2 , E_3 , and E_4 , the treatments that reduces the severity are T_2 and T_3 , while T_4 and T_5 reduce likelihood.

However, it could be the case that some treatment in $rel\_treatments$ has an overlapped effect in reducing risks with other treatments. Thus, we evaluate each subset of $rel\_treatments$ whether it is adequate to mitigate the risks, such that they (i.e., $candidate\_solution$, subset of $rel\_treatments$) satisfy the evidence values specified in $desired\_labels$ and acc_risks. Then, the treatments and the input goals can be added to the $solution$ and the cost is calculated (Algorithm 1 line 13).

$Forward\_Reasoning$ (Algorithm 3), essentially, is the adaptation of the one proposed in [13]. The algorithm consists of two main loops. The first loop propagates the input evidence throughout the GR model updating the nodes' labels without considering alleviation relations (line 5-10). $Update\_Label$ (Algorithm 4) updates SAT and DEN following the relation defined in Tab. 1 (for decomposition and contribution relations line 6-7) and Tab. 3 (for impact relation line 3-4). Based on this result and looking at the evidence values, we can identify which treatments are adopted. Thus, using $Apply\_Alleviation$ (Algorithm 5) we rewrite the sign of impact relations that are mitigated by treatments – the rules (Tab. 4) are encoded in the line 3 of the algorithm. Afterwards, the rewritten GR model ($\langle \mathcal{N}, \mathcal{R} \rangle$) is again evaluated in the second loop. Here, the final values for all nodes are calculated using all relations, including the alleviation ones.

[5] $Related$ is meant to enumerate all the nodes that is reachable from a certain node in $\langle \mathcal{N}, \mathcal{R} \rangle$

Algorithm 3 Forward_Reasoning

Require: goal_model $\langle \mathcal{N}, \mathcal{R} \rangle$, label_array initial
label_array current, old
current ←initial
i ←0
repeat
  while $old \neq current$ **do**
    old ←current
    for all $N_i \in \mathcal{N} \ \wedge \neg is\_Impact(N_i)$ **do**
      current[i] ←$Update\_Label(i, \langle \mathcal{N}, \mathcal{R} \rangle, old)$
    end for
  end while
  if i=0 **then**
    $Apply\_Alleviation(\langle \mathcal{N}, \mathcal{R} \rangle, current)$
    old ←**nil**
  end if
  $i++$
until i=2
return current

Algorithm 4 Update_Label

Require: int i, goal_model $\langle \mathcal{N}, \mathcal{R} \rangle$, label_array old
for all $R_j \in \mathcal{R}$ s.t. $target(R_j) = N_i$ **do**
  if $is\_Impact(R_j)$ **then**
    sat$_{ij}$ = $Apply\_Imp\_Sat(N_i, R_j, Old)$
    den$_{ij}$ = $Apply\_Imp\_Den(N_i, R_j, Old)$
  else {decomposition and contribution relations}
    sat$_{ij}$ = $Apply\_Rules\_Sat(N_i, R_j, Old)$
    den$_{ij}$ = $Apply\_Rules\_Den(N_i, R_j, Old)$
  end if
end for
return $\{max(max\_array(sat_{ij}), Old[i].sat)$, $max(max\_array(den_{ij}), Old[i].den)\}$

Algorithm 5 Apply_Alleviation

Require: goal_model $\langle \mathcal{N}, \mathcal{R} \rangle$, label_array current
for all $N_k \in \mathcal{N} \ \wedge \rangle \int \mathcal{I} \Updownarrow \surd \dashv \rfloor \sqcup \langle \mathcal{N}_{\|} \rangle$ **do**
  for all $R_l \in \mathcal{R} \ \wedge \rangle \int \mathcal{A} \updownarrow \rceil \sqsubseteq \rangle \dashv \sqcup \rangle \wr \backslash (\mathcal{R}_{\uparrow})$ **do**
    N$_k$ ←$Update\_Sign(R_l, current)$
  end for
end for

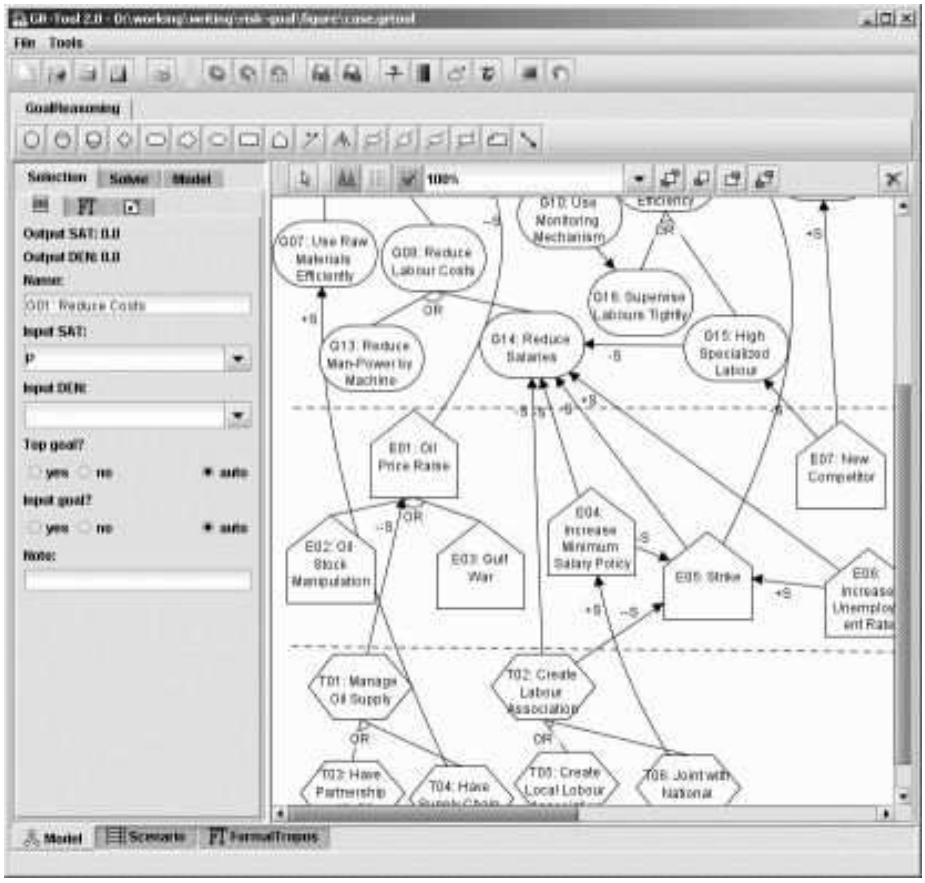

Figure 3: Goal Risk Tool

6 GR-Tool and Experimental Results

The tool we have developed is an extension of the Goal Reasoning Tool[6] (GR-Tool) developed within the Tropos project. Basically, the tool (Fig. 3) is a graphical tool in which it is possible to draw GR models and run algorithms presented in the previous section. The algorithms have been fully implemented in JAVA and are embedded in the tool.

To test our approach and its implementation, we run a number of experiments with the *loan origination process* case study that we summarize briefly in the following.

Let's start from the situation where we are interested to obtain *partial* evidence for the satisfaction of top-goals earn more income *(G_1)* and ensure repayment of loan *(G_7)*, while *full* evidence for top-goals receive loan application *(G_4)* and handle loan application *(G_{10})*. Suppose also, that the maximum level of risk we are willing to run is $Den(G_4) = P$,$Den(G_{10}) = P$,$Den(G_1) = N$, and $Den(G_7) = N$. Given these inputs, the set of possible solutions is reported in Tab. 5. Note that these solutions do not consider risk for the moment. The total cost of each solution is calculated summing up the cost of each leaf goal (input goal). Among these solutions,

[6] http://sesa.dit.unitn.it/goaleditor/

Input Goal	Cost	S1	S2	S3	S4	S5	S6	S7	S8
G02-Earn from Loan Interest	3	X		X		X		X	
G03-Charge High Fee for LOP	2		X		X		X		X
G05-Receive Hard-copy App.	5	X	X			X	X		
G06-Receive Electronic App.	3			X	X			X	X
G08-Ask Mortgage	2	X	X	X	X	X	X	X	X
G09-Monitor Usage of Loan	4	X	X	X	X	X	X	X	X
G11-Verify Loan Application	3	X	X	X	X	X	X	X	X
G13-Assessed by CB	10	X	X	X	X				
G14-Assessed by In-house	8					X	X	X	X
G16-Proposed by Customer	1	X	X	X	X	X	X	X	X
G17-Defined by Bank	3	X	X	X	X	X	X	X	X
G19-Approved by Clerk	1	X	X	X	X	X	X	X	X
G20-Approved by Manager	1	X	X	X	X	X	X	X	X
Cost		33	32	31	30	31	30	29	28

Table 5: Cost of Alternative Solutions

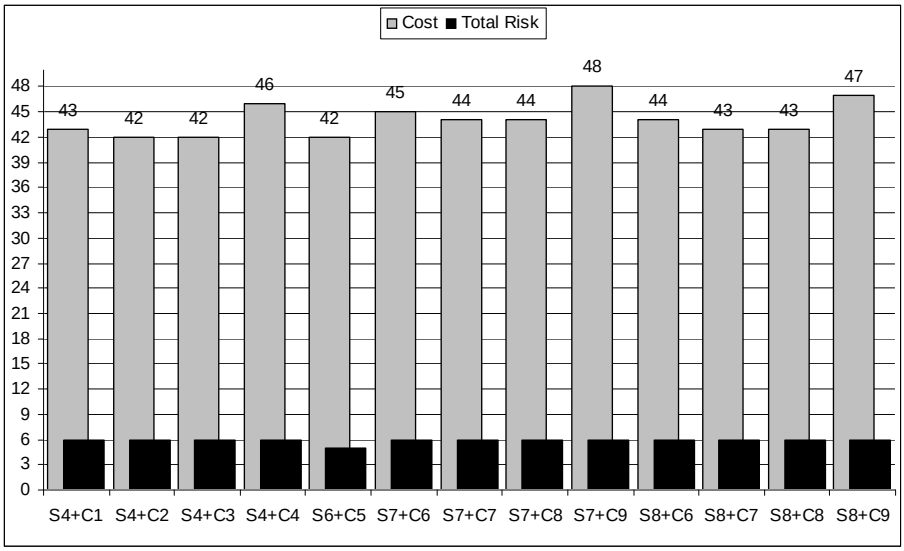

Figure 4: Comparison Total Risk and Total Cost among All Candidate Alternatives and Their Treatments

suppose we decide to focus on solutions with a cost lesser than 30, that are S4, S6, S7, and S8. Particularly, let's consider S4 with the initial assignment for input goals as reported in Tab. 6 (column "Goal-In"). This assignment satisfies the desired values for top-goals (column "Goal-Out"). Now if we introduce the assignment to events reported in column "Event-In", the desired values for top goals are not anymore satisfied ("Event-Out"). For instance, G_4 has *full* DEN evidence, while the acceptable value was at most *partial*. This happens since goal receive loan application *(G_4)* is satisfied by receive electronic application *(G_6)* goal, which has *full* DEN evidence (i.e., $Den(G_6) = F$). We have a similar situation for G_7 and G_{10} . To make S4 acceptable, possible sets of treatments are C1, C2, C3, and C4 in Tab. 7. As reported in column "Treat-Out" in Tab. 6, C1 satisfies again the desired values for top-goals (DEN values for G_4 and G_{10} are now *partial*, while goal G_7 has no evidence for denial). However, the adoption of C1 introduces an additional cost for S4 that is now 30+13=43.

	Goal			Event			Treat.		
	In	Out		In	Out		In	Out	
	S	S	D	S	S	D	S	S	D
E01-Increase Interest Rate of Loan	-	-	-	-	-	-	-	-	-
E02-Forge Electronic Application	-	-	-	-	F	-	-	F	-
E03-Forgery from External Attack	-	-	-	F	F	-	F	F	-
E04-Forgery from Internal Breach	-	-	-	F	F	-	F	F	P
E05-Cust Fails to Fulfill Loan Sch	-	-	-	-	P	-	-	P	-
E06-Fake Identity Document	-	-	-	F	F	-	F	F	F
E07-Fake Application	-	-	-	P	P	-	P	P	-
E08-Credit Bureau Ignorance	-	-	-	P	P	-	P	P	-
E09-Mispredict Monetary Cond.	-	-	-	F	F	-	F	F	-
E10-Collusion Customer-Clerk	-	-	-	P	P	-	P	P	-
E11-Cust. Unemployment	-	-	-	P	P	-	P	P	-
E12-Economic Crisis	-	-	-	-	-	-	-	-	-
G01-Earn More Income	-	P	-	-	P	-	-	P	-
G02-Earn from Loan Interest	-	-	-	-	-	-	-	-	-
G03-Charge High Fee for LOP	P	P	-	P	P	-	P	P	-
G04-Receive Loan Application	-	F	-	-	F	**F**	F	F	**P**
G05-Receive Hard-copy App.	-	-	-	-	-	-	-	-	-
G06-Receive Electronic App.	F	F	-	F	F	**F**	F	F	**P**
G07-Ensure Repayment of Loan	-	P	-	-	P	**P**	-	P	**-**
G08-Ask Mortgage	F	F	-	F	F	-	F	F	-
G09-Monitor Usage of Loan	P	P	-	P	P	-	P	P	-
G10-Handle Loan Application	-	F	-	-	F	**F**	-	F	**P**
G11-Verify Loan Application	F	F	-	F	F	**F**	F	F	**P**
G12-Assess Application	-	F	-	-	F	**P**	-	F	-
G13-Assessed by Credit Bureau	F	F	-	F	F	**P**	F	F	-
G14-Assessed by In-house	-	-	-	-	-	-	-	-	-
G15-Define Loan Schema	-	F	-	-	F	P	-	F	P
G16-Proposed by Customer	F	F	-	F	F	P	F	F	P
G17-Defined by Bank	F	F	-	F	F	-	F	F	-
G18-Approve Loan Application	-	F	-	-	F	-	-	F	-
G19-Approved by Clerk	F	F	-	F	F	-	F	F	-
G20-Approved by Manager	F	F	-	F	F	-	F	F	-
T01-Use Digital Signature	-	-	-	-	-	-	-	P	-
T02-Have Digital Signature Inf.	-	-	-	-	-	-	P	P	-
T03-Install Public Key Inf.	-	-	-	-	-	-	P	P	-
T04-Employ Intrusion Det. Sys.	-	-	-	-	-	-	-	-	-
T05-Employ Strict Access Control	-	-	-	-	-	-	-	P	-
T06-Hire Underwriter	-	-	-	-	-	-	P	P	-
T07-Verify ID doc with Gov. DB	-	-	-	-	-	-	F	F	-
T08-Assign Liaison Officer for CB	-	-	-	-	-	-	P	P	-
T09-Train Internal Actuary	-	-	-	-	-	-	-	-	-
T10-Assess App. Anonymously	-	-	-	-	-	-	-	-	-

Table 6: SAT-DEN Values of S4-alternative and C1-treatments

		S4				S6	S7/S8			
Treatment	Cost	C1	C2	C3	C4	C5	C6	C7	C8	C9
T02	2	X			X		X			X
T03	2	X		X	X		X		X	X
T04	1		X	X	X			X	X	X
T05	2		X		X			X		X
T06	4	X	X	X	X	X	X	X	X	X
T07	3	X	X	X	X	X	X	X	X	X
T08	2	X	X	X	X					
T09	3					X	X	X	X	X
T10	2					X	X	X	X	X
Total Cost		13	12	12	16	12	16	15	15	19

Table 7: Cost of Possible Treatments

A similar analysis can be done for the other selected solutions S6, S7 and S8 (i.e., those with a cost lesser than 30). The set of treatments for S6 is C5, while for S7 and S8 can be either C6, C7, C8, or C9. Their costs are reported in Tab. 7.

Fig. 4 shows the comparison among costs and risks for all solutions and related treatments. The *total risk* is calculated assuming Null=1, Partial=2, and Full=3 and summing up the DEN values for all top goals. This means that for the acceptable risk level (i.e., $Den(G_1) = N$,$Den(G_4) = P$,$Den(G_7) = N$, and $Den(G_{10}) = P$), we can have at most the total risk 1+1+2+2=6. Note that S6+C5 has a lower total risk w.r.t. the others (i.e., C6+C5 total risk=5) and is cheaper than the initial S4+C1 we considered. So S6+C5 seems to be the most convenient solution to be adopted. However, the consequence of adopting S6+C5 is that the customers cannot submit their loan application electronically, and the analyst should consider this in the choice.

7 Conclusions and Future Work

In this paper, we have presented a framework to model and reason about risk within the requirements engineering process. We have adopt and extended the Tropos goal modeling framework and proposed qualitative reasoning algorithms to analyze risk during the process of evaluation and selection of alternatives.

Our approach has some limitations that we would like to overcome in our future work. Particularly, the fact that we combine evidence values of satisfaction and denial using simple maximum and minimum metrics does not allow us to represent the difference among countermeasures that are adopted in the solution. For instance, the solution S4+C1 has the same level of risk of S4+C4, but S4+C4 adopts more treatments than the former. Moreover, even if the three qualitative values used for SAT and DEN can be extended including more intermediate values, as we done for Tropos goal models we would like to propose quantitative reasoning mechanisms where evidence is expressed in term of probability.

Acknowledgment

This work has been partly supported by the projects EU-SERENITY, FIRB-ASTRO, PAT-MOSTRO, PAT-STAMPS.

References

[1] *Military Standard, Procedures for Performing a Failure Mode, Effects, and Critical Analysis*. Number MIL-STD-1692A. U.S. Department of Defense, 1980.

[2] A. I. Anton. Goal-Based Requirements Analysis. In *Proceedings of the 2nd IEEE International Conference on Requirements Engineering (ICRE'96)*, page 136, Washington, DC, USA, 1996. IEEE Computer Society Press.

[3] T. Bedford and R. Cooke. *Probabilistic Risk Analysis: Foundations and Methods*. Cambridge University Press, 2001.

[4] B. W. Boehm. Software Risk Management: Principles and Practices. *IEEE Software*, 8(1):32–41, 1991.

[5] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos. Tropos: An Agent-Oriented Software Development Methodology. *Journal of Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, 2004.

[6] S. A. Butler. Security Attribute Evaluation Method: a Cost-Benefit Approach. In *Proceedings of the International Conference on Software Engineering (ICSE'02)*, pages 232–240, New York, NY, USA, 2002. ACM Press.

[7] M. J. Carr, S. L. Konda, I. Monarch, F. C. Ulrich, and C. F. Walker. Taxonomy-Based Risk Identification. Technical Report CMU/SEI-93-TR-6, ESC-TR-93-183, Software Engineering Institute, Carnegie Mellon University, June 1993.

[8] L. K. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos. *Non-Functional Requirements in Software Engineering*. Kluwer Academic Publishers, 2000.

[9] S. L. Cornford, M. S. Feather, V. A. Heron, and J. S. Jenkins. Fusing Quantitative Requirements Analysis with Model-based Systems Engineering. In *Proceedings of the 14th IEEE International Requirements Engineering Conference*, pages 279–284, Los Alamitos, CA, USA, 2006. IEEE Computer Society Press.

[10] A. Dardenne, A. van Lamsweerde, and S. Fickas. Goal-Directed Requirements Acquisition. *Science of Computer Programming*, 20:3–50, 1993.

[11] F. den Braber, T. Dimitrakos, B. A. Gran, M. S. Lund, K. Stølen, and J. Ø. Aagedal. The CORAS Methodology: Model-Based Risk Assessment using UML and UP. In *UML and the Unified Process*, pages 332–357. Idea Group Publishing, 2003.

[12] M. S. Feather. Towards a Unified Approach to the Representation of, and Reasoning with, Probabilistic Risk Information about Software and its System Interface. In *Proceedings of the 15th IEEE International Symposium on Software Software Reliability Engineering*, pages 391–402. IEEE Computer Society Press, November 2004.

[13] P. Giorgini, J. Mylopoulos, E. Nicchiarelli, and R. Sebastiani. Formal Reasoning Techniques for Goal Models. *Journal of Data Semantics*, 1(1):1–20, October 2003.

[14] P. Giorgini, J. Mylopoulos, and R. Sebastiani. Goal-Oriented Requirements Analysis and Reasoning in the Tropos Methodology. *Engineering Applications of Artificial Intelligence*, 18(2):159–171, March 2005.

[15] G. Helmer, J. Wong, M. Slagell, V. Honavar, L. Miller, and R. Lutz. A Software Fault Tree Approach to Requirements Analysis of an Intrusion Detection System. *Requirements Engineering Journal*, 7(4):207–220, 2002.

[16] G. A. Holton. Defining Risk. *Financial Analyst Journal*, 60(6):1925, 2004.

[17] J. Jürjens. Towards Secure Systems Development with UMLsec. In *Proceedings of the 4th International Conference on Fundamental Approaches to Software Engineering*, pages 187–200. Springer-Verlag, 2001.

[18] T. A. Kletz. HAZOP - Past and Future. *Reliability Engineering and System Safety*, 55(3):263–266, March 1997.

[19] S. Lee, R. Gandhi, and G. Ahn. Security Requirements Driven Risk Assessment for Critical Infrastructure Information Systems. In *Proceedings of the 3rd Symposium on Requirements Engineering for Information Security*, 2005.

[20] L. Liu, E. S. K. Yu, and J. Mylopoulos. Security and Privacy Requirements Analysis within a Social Setting. In *Proceedings of the 11th IEEE International Requirements Engineering Conference*, pages 151–161, 2003.

[21] T. Lodderstedt, D. Basin, and J. Doser. SecureUML: A UML-Based Modeling Language for Model-Driven Security. In *Proceedings of the 5th International Conference on the Unified Modeling Language – the Language and its Applications*, volume 2460 of *Lecture Notes in Computer Science*, pages 426–441. Springer-Verlag, 2002.

[22] R. R. Lutz and R. M. Woodhouse. Requirements Analysis Using Forward and Backward Search. *Annals Software Engineering*, 3:459–475, 1997.

[23] N. Mayer, A. Rifaut, and E. Dubois. Towards a Risk-Based Security Requirements Engineering Framework. In *Proceedings of the 11th International Workshop on Requirements Engineering: Foundation for Software Quality*, 2005.

[24] J. McDermott and C. Fox. Using Abuse Case Models for Security Requirements Analysis. In *Proceedings of 15th Annual Computer Security Applications Conference*, pages 55–64, Phoenix, AZ, USA, 1999.

[25] C. P. Pfleeger and S. L. Pfleeger. *Security in Computing*. Prentice-Hall, 4th edition, 2006.

[26] G. G. Roy and T. L. Woodings. A Framework for Risk Analysis in Software Engineering. In *APSEC '00: Proceedings of the Seventh Asia-Pacific Software Engineering Conference*, page 441, Washington, DC, USA, 2000. IEEE Computer Society Press.

[27] B. Schneier. Attack Trees: Modeling Security Threats. *Dr. Dobbs Journal*, 12(24):21–29, 1999.

[28] R. Sebastiani, P. Giorgini, and J. Mylopoulos. Simple and Minimum-Cost Satisfiability for Goal Models. In *Proceedings of the 16th Conference On Advanced Information Systems Engineering*, volume 3084 of *Lecture Notes in Computer Science*, pages 20–33. Springer-Verlag Heidelberg, June 2004.

[29] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, NJ, 1976.

[30] B. Shawn and F. Paul. Multi-Attribute Risk Assessment. Technical Report CMU-CS-01-169, Carnegie Mellon University, December 2001.

[31] G. Sindre and A. L. Opdahl. Eliciting Security Requirements With Misuse Cases. *Requirements Engineering Journal*, 10(1):34–44, Jan. 2005.

[32] M. Stamatelatos, W. Vesely, J. Dugan, J. Fragola, J. Minarick, and J. Railsback. *Fault Tree Handbook with Aerospace Applications*. NASA, 2002.

[33] US-Department of Defense. *Department of Defense Information Technolgoy Security Certification and Accreditation Process (DITSCAP) Application Manual*, July 2000.

[34] A. van Lamsweerde, S. Brohez, R. D. Landtsheer, and D. Janssens. From System Goals to Intruder Anti-Goals: Attack Generation and Resolution for Security Requirements Engineering. In *Proceedings of the 2nd International Workshop on Requirements for High Assurance Systems*, 2003.

[35] A. van Lamsweerde and E. Letier. Handling Obstacles in Goal-Oriented Requirements Engineering. *IEEE Transactions on Software Engineering*, 26(10):978–1005, 2000.

[36] E. Yu. *Modelling Strategic Relationships for Process Engineering*. PhD thesis, University of Toronto, Department of Computer Science, 1995.