Requirements Eng (2011) 16:101–116
DOI 10.1007/s00766-010-0112-x

ORIGINAL RESEARCH

Goal-driven risk assessment in requirements engineering

Yudistira Asnar · Paolo Giorgini · John Mylopoulos

Received: 9 March 2010 / Accepted: 9 August 2010 / Published online: 11 September 2010

Abstract Risk analysis is traditionally considered a critical activity for the whole software system's lifecycle. Risks are identified by considering technical aspects (e.g., failures of the system, unavailability of services, etc.) and handled by suitable countermeasures through a refined design. This, however, introduces the problem of reconsidering system requirements. In this paper, we propose a goal-oriented approach for analyzing risks during the requirements analysis phase. Risks are analyzed along with stakeholder interests, and then countermeasures are identified and introduced as part of the system's requirements. This work extends the Tropos goal modeling formal framework proposing new concepts, qualitative reasoning techniques, and methodological procedures. The approach is based on a conceptual framework composed of three main layers: assets, events, and treatments. We use "loan origination process" case study to illustrate the proposal, and we present and discuss experimental results obtained from the case study.

Keywords Risk assessment · Requirement analysis · Goal-oriented requirement engineering

Y. Asnar (✉) · P. Giorgini · J. Mylopoulos
Department of Information Engineering and Computer Science,
University of Trento, Trento, Italy
e-mail: yudis.asnar@disi.unitn.it

P. Giorgini
e-mail: paolo.giorgini@disi.unitn.it

J. Mylopoulos
e-mail: jm@disi.unitn.it

1 Introduction

Traditionally, risk analysis is used in software development to identify situations or events that can cause project failures. It offers methods and techniques for documenting the impact of mitigation strategies [35] and for evaluating system criticality [9]: risks are analyzed and mitigating countermeasures are then introduced. Introducing countermeasures corresponds, in the best case, to fine-tuning of the initial design. Unfortunately, in many cases risk mitigation requires the revision of the entire design and, possibly, of initial requirements. Shifting risk analysis to the early phases of the software development process can give the obvious advantage of considering risk mitigations as an integral part of the initial design. Along this direction, there have been several recent attempts to integrate risk analysis with requirement analysis [16]. The idea is to analyze risks along with stakeholder needs and to introduce risk-based criteria for choosing among alternative ways of fulfilling requirements.

According to Goal-Oriented Requirements Engineering, analysis of stakeholder goals leads to alternative sets of functional requirements that can each fulfill these goals. These alternatives can be evaluated with respect to non-functional requirements posed by stakeholders. KAOS [17], *i** [44], GBRAM [1], and Tropos [11] are examples of goal-oriented methodologies and frameworks that have gained popularity. In particular, *i** proposes a modeling and analysis technique for early requirements, where the analyst identifies 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 actor dependencies, one can answer *why* questions, besides *what* and *how*, regarding system functionalities/requirements. Answers to *why* questions ultimately link system functionalities to stakeholder needs. Moreover, *i** analyzes goals through a refinement process where each goal is decomposed into subgoals and positive/negative contributions are established among goals. The result of the analysis is a goal model that characterizes alternative ways of fulfilling the top-level goals. For example, in the case of a loan origination process, the goal of `assess loan application` can be OR-decomposed into `in-house assessment` and `Credit Bureau assessment`, where Credit Bureau is a company that provides credit information and assesses loan applications. Likewise, the goal `receive electronic application` may help (i.e., contributes positively to) `verify loan application`, because the bank or credit bureau can indeed send the application electronically, and they can also validate it semi-automatically by using appropriate software. In [19, 37], the authors proposed an extension of *i**, namely the Tropos Goal Modeling, with formal semantics for automatic reasonings.

In this paper, we propose a goal-oriented framework (Goal-Risk framework—GR, for short) for modeling and reasoning about risk during requirements analysis extending the Tropos goal modeling framework. It is introducing a three-layer analysis model founded on three main concepts: *asset, event*, and *treatment*. Assets[1], modeled in terms goals, are analyzed and related to external events that can influence negatively (i.e., risk) their satisfaction. Treatments are then introduced to mitigate the effects of such events. Besides modeling inter-relationships among layers (adopted from [18]), the GR framework supports the representation of relationships within each layer. For instance, the goal `offer low-interest loans` may contradict with the goal of `have high profits`; or in the event layer, the event `economic recession` may make the event `debtor defaults on loan` more likely. These intra-layer dependencies make risk analysis more accurate. To support the analysis process, we propose qualitative risk reasoning techniques intended to identify designs that minimize risk while fulfilling requirements. The main contributions of this paper are: proposing a modeling framework to capture and analyze risks during requirement engineering phase and the framework supports modeling, analyzing, and assessing risks along requirement analysis by extending the expressivity of Defect, Detection and Prevention (DDP) model [16]. This framework is composed of a modeling framework, a methodological process to develop and analyze the model, several analytical techniques, and supporting tools. The end results of this framework are a set of requirements that realize stakeholders' intentions and mitigate relevant risks into acceptable levels.

The rest of the paper is organized as follows. In Sect. 2, we present related work to the GR framework Sect. 3 introduces the *Loan Origination Process* scenario that is used to describe the GR analysis framework in Sect. 4. The risk assessment process and algorithms for qualitative reasoning are presented in Sect. 5, while in Sect. 6, we describe the CASE tool we have developed, along some experimental results. Section 7 offers a discussion of pros and cons of the framework, and conclusions are presented in Sect. 8.

2 Related work

Research on three major areas are related to our work: Requirements Engineering, Secure and Dependable Engineering, and Risk Analysis.

In **Requirements Engineering**, Dardenne et al. [17] 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*. In later work, KAOS introduces also the concept of *obstacle* [25] and *anti-goal* [26] in order to analyze boundary conditions and failure situations for a design. Obstacles are situations that can lead to goal failure. Anti-goals, on the other hand, are goals associated with malicious stakeholders, such as an attacker. In other words, obstacles are unintended risks, while anti-goals are threats or intended risks. These features make KAOS suitable for analyzing requirements for secure and/or dependable systems. In [25], van Lamswerdee and Letier present a collection of techniques for deriving obstacles systematically from goals and domain properties, also for resolving. Mayer et al. [31, 32] extends the *i** modeling framework [44] to analyze risk and security issues during requirement analysis. The framework models business assets (including business goals) of an organization and assets of its IT systems (such as architectures and code). Countermeasures are then selected to mitigate risks, thereby ensuring that risks will not affect any assets. Liu et al. [28] propose a methodological framework for security requirements analysis founded on *i** and the NFR framework [15]. In particular, their analysis explores alternative designs and evaluates them on the basis of threats, vulnerabilities, and countermeasures.

In the area of **Secure and Dependable Systems**, the most popular analysis frameworks are Fault Tree Analysis (FTA) [42], Failure Modes, Effects, and Criticality Analysis (FMECA) [43]. In security engineering, approaches such as *attack trees* and *threat trees* [20, 36] are similar to

[1] In ISO 13335, asset is defined as "anything that has value to an organization". As such, assets may be (1) resources, (2) tasks executed to generate value, and (3) targets/objectives/goals whose fulfillment generates value. This paper concentrates on analyzing assets as goals.

FTA, while other proposals such as UMLSec [23], SecureUML [29], Abuse Case [33], and Misuse Case [40] constitute UML extensions intended to deal specifically with security concerns. The most relevant work for our purposes is the Defect Detection and Prevention (DDP) proposal by Feather et al. [18]. DDP consists of a three-layer model consisting, respectively of Objectives, Risks, and Mitigations. Each objective has a *weight* to represent its importance, each risk has a *likelihood* of occurrence, while every mitigation has a *cost* for its accomplishment (mostly resource consumption). Severity of a risk can be represented by an impact relationship between an objective and a risk. Moreover, a DDP model specifies how to compute the level of objective achievements and the cost of mitigations. This calculation allows one to evaluate the impact of a collection of countermeasures, thereby supporting risk analysis. The DDP model can be integrated with other quantitative frameworks (e.g., FMECA, FTA) in order to model and assess risks/failures [18]. All the works mentioned above propose analysis techniques (quantitative and/or qualitative) to assess failures (or events in the GR framework). The DPP framework constitutes a baseline for our work on the GR framework.

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) [8] is widely used for quantitative risk assessment, while approaches like FMECA [43] quantify risk into qualitative values: frequent, reasonably probable, occasional, remote, and extremely unlikely. Events are prioritized using the notion of "expectancy loss" resulted from them which is defined as the product of its likelihood and severity. Priority here reflects the criticality of an event. When resources are limited, an analyst may decide to adopt countermeasures for mitigating events on the basis of their priority. However, estimation of probabilities is generally imprecise, as they typically strongly depend on expert judgment. Approaches such as Multi-Attribute Risk Assessment [12] can improve the risk assessment process by considering multi-attribute analysis. In this work, many factors that can impact the quality of a system—such as reliability, availability, safety and confidentiality—are analyzed for potential risks. For instance, an Air Traffic Management system is required to always be available and safe. Certain conditions (e.g., radar noise) can affect the normal behavior of the system and consequently impact on its safety. In many cases, the best way to deal with radar noise is to restart the system. This, however, impacts on its availability. This inter-dependence of quality factors introduces the need for analysis that finds the right tradeoffs. In [13], Butler proposes a technique for selecting cost-effective countermeasures to deal with existing security threats by using multi-attribute risk assessment. This line of research has identified important properties of risk, such as likelihood and severity, which must be accounted for in the GR framework. Finally, CORAS [10] is a framework for risk analysis of security-critical systems. The CORAS risk management process consists of the following steps: context identification, risk identification, risk analysis, risk evaluation, and risk treatment. Moreover, CORAS includes a modeling language defined in terms of a UML profile.

3 Running example

The scenario we present below originated within the European project SERENITY[2], and focuses on a typical *Loan Origination Process* (LOP) for a bank. The process starts when a loan application is received, and ends with a decision. Clearly, the bank aims to `earn income`, `accept loan applications`, `handle the applications`, and `ensure loan repayment`. When the bank receives the application, it starts the process of verifying data and calculating credit rating for the applicant. The rating may be generated internally (in-house assessment) or externally from a Credit Bureau. Next, the bank decides on a loan scheme consisting of a loan cap and the interest to be paid. We assume that a loan scheme is initially proposed by the customer, while the bank makes the final decision. The bank is, of course, also interested in ensuring the repayment of the loan in order to increase its income.

Several events (i.e., threats and/or unintended happenings) may endanger the success of the whole process. For instance, `forgery of the loan application`, `fake identification documents`, `inaccurate credit rating` are potential dangers for the LOP. Accordingly, analysts need to assess their potential impact. If this is deemed unacceptable, then the designer may want to introduce measures aiming at reducing the likelihood, or mitigating the effects of these events. Of course, these additions have to be analyzed carefully before their adoption because they introduce additional cost and delay in the origination process, or even can introduce new risks.

4 Tropos goal risk framework

Tropos is a software development methodology that adopts the concepts of agent goal, task, and resource and uses them throughout the development process [11], from early requirements analysis to implementation. Early requirements analysis model and analyze the organizational setting where the system-to-be will eventually operate. In the following, we extend the Tropos goal modeling framework

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

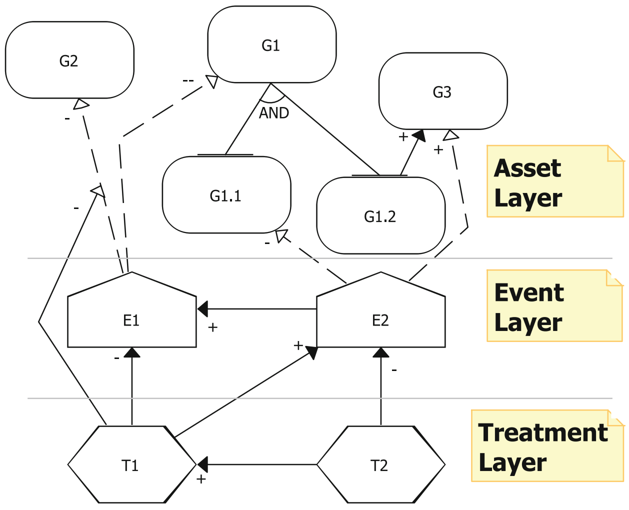


Fig. 1 Goal-risk model

[19, 37] by introducing constructs and relations specific for analyzing risk.

Conceptually, a Goal-Risk (GR) model (see Fig. 1) consists of three layers representing assets, events, and treatments. A GR model is defined mathematically as a triplet $\langle \mathcal{N}, \mathcal{R}, \mathcal{I} \rangle$, where $\mathcal{N}$ is a set of nodes, $\mathcal{R}$ a set of relations among nodes, and $\mathcal{I}$ represents a special impact relation that depicts the severity of an event affecting the asset layer. Impact relations, depicted as dash line-arrows, are special because on one hand they are relations between the event layer and the asset layer, and on the other they are the target for alleviation relations. We distinguish the severity into 4 levels: $+$, $++$, $-$, and $--$, where $++$ and $--$ are stronger than $+$ and $-$, respectively.

$\mathcal{N}$ is comprised of three types of constructs: *goals, tasks*, and *events*. Goals (depicted as ovals) are strategic interests that stakeholders/actors intend to achieve for generating values. Events (depicted as pentagons) are uncertain circumstances, typically out of the control of actors, that can have an impact (positive or negative) on the fulfillment of goals. Tasks (depicted as hexagons) are sequences of actions used to achieve goals or to treat events.[3] Each construct has two attributes: $SAT - Sat(N)$ and $DEN - Den(N)$. Such attributes represent, respectively, available evidence that the construct N will be satisfied/fulfilled/present or denied/failed/absent. In probability theory, if $\text{Prob}(A) = 0.1$ then 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 [38], the evidence of a goal being denied (DEN) cannot be inferred from evidence on the satisfaction of the goal (SAT), and vice versa. For instance, the bank has the goal `verify loan application`, and the goal is affected by the event `fake identity document`. The only conclusion we can draw from this is DEN, while we cannot say anything about SAT since there is no information on the satisfaction of the goal. 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 goal means that there is (at least) "sufficient" ("some", "no") evidence to support the claim that the goal will 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 claim that the goal will be denied.

Relations $\mathcal{R}$ are represented as $(N_1, \ldots, N_n) \overset{r}{\longmapsto} N$, 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*, and *alleviation* relations. AND/OR decomposition relations are used to refine goals, tasks, and events into finer-grain models. Contribution relations (depicted as solid line with filled arrow) are used to model the impact of a node over another. Our framework distinguishes four levels of contribution relations: $+$, $++$, $-$, and $--$. Each one of these types 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 toward target nodes. The same intuition is applied for the other types of contribution in delivering DEN evidence. However, the "$--$" and "$-$" propagate bipolar evidence. For instance, "$-_S$" propagates the SAT value of the source node to the DEN value of the target node, and the same principle holds for others. Alleviation relations (depicted as solid lines with hollow arrows) have a similar definition with contribution relations, but a slight different semantics (as we shall see later). These relations relate treatments with impact relations (i.e., nodes to relations) to model severity reduction of impact relations by treatments. In the following subsections, we describe the three layers of the GR model through the loan origination process scenario together with the formal semantics of each relation.

4.1 Asset layer

The asset layer is adopted from the Tropos goal model [19] which analyzes strategic interests of stakeholders. In this layer of analysis, the goals of stakeholders are identified, refined, and analyzed along with inter-relationships among them. As shown in Fig. 2, modeling starts by identifying top stakeholder goals. For bank management, these are `earn more income` (G_1), `receive loan application` (G_4), `ensure loan repayment` (G_7), and `handle loan application` (G_{10}).

[3] In this paper, we use hexagons **only** to denote an event treatment/countermeasure.

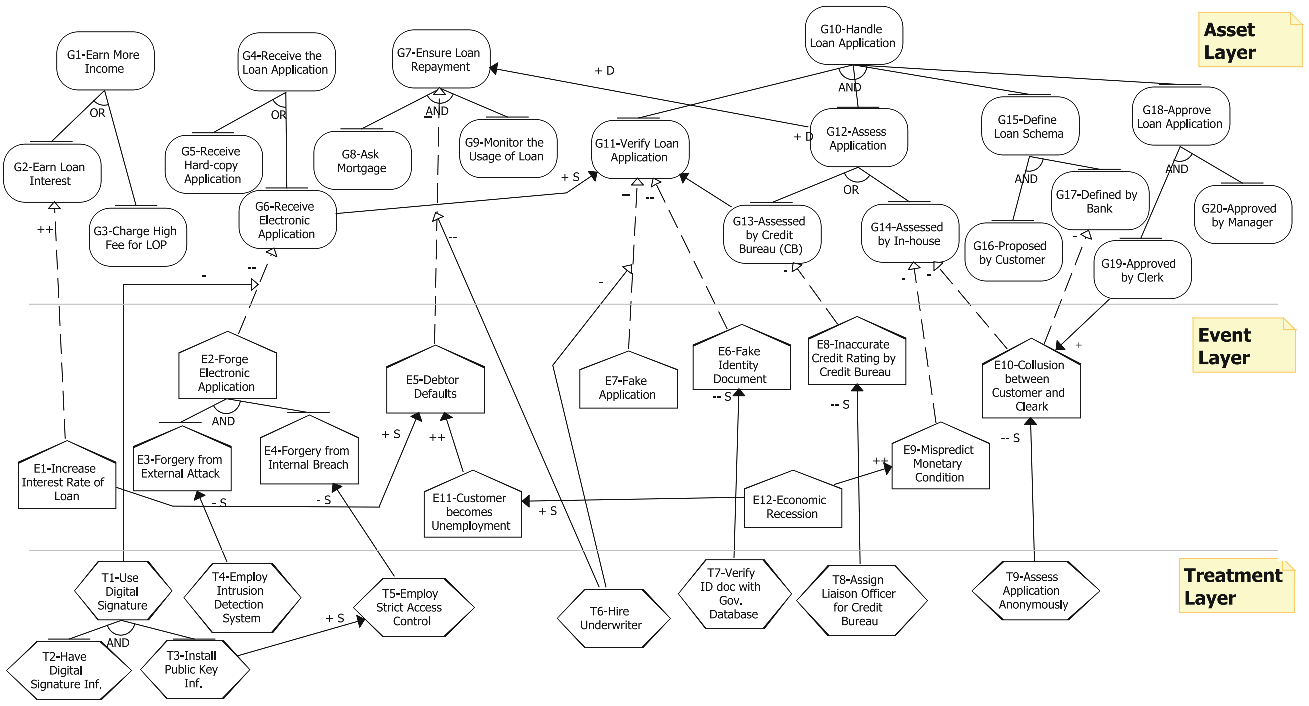

Asset Layer
Event Layer
Treatment Layer
G1-Earn More Income
OR
G2-Earn Loan Interest
G3-Charge High Fee for LOP
G4-Receive the Loan Application
OR
G5-Receive Hard-copy Application
G6-Receive Electronic Application
G7-Ensure Loan Repayment
AND
G8-Ask Mortgage
G9-Monitor the Usage of Loan
G10-Handle Loan Application
AND
G11-Verify Loan Application
G12-Assess Application
OR
G13-Assessed by Credit Bureau (CB)
G14-Assessed by In-house
G15-Define Loan Schema
AND
G16-Proposed by Customer
G17-Defined by Bank
G18-Approve Loan Application
AND
G19-Approved by Clerk
G20-Approved by Manager
+ D
+ D
+ S
++
--
-
E1-Increase Interest Rate of Loan
E2-Forge Electronic Application
AND
E3-Forgery from External Attack
E4-Forgery from Internal Breach
E5-Debtor Defaults
E6-Fake Identity Document
E7-Fake Application
E8-Inaccurate Credit Rating by Credit Bureau
E9-Mispredict Monetary Condition
E10-Collusion between Customer and Cleark
E11-Customer becomes Unemployment
E12-Economic Recession
+ S
++
+ S
++
+
-- S
-- S
-- S
- S
- S
T1-Use Digital Signature
AND
T2-Have Digital Signature Inf.
T3-Install Public Key Inf.
T4-Employ Intrusion Detection System
T5-Employ Strict Access Control
+ S
T6-Hire Underwriter
T7-Verify ID doc with Gov. Database
T8-Assign Liaison Officer for Credit Bureau
T9-Assess Application Anonymously

Fig. 2 Goal-risk model for the loan origination process

Table 1 Evidence propagation rules for (AND/OR) decomposition and contribution relations

Relation	$\mathrm{Sat}(N_1)$	$\mathrm{Den}(N_1)$
$(N_2,N_3)\overset{\text{and}}{\longmapsto}N_1$	$\min\left\{\begin{matrix}\mathrm{Sat}(N_2),\\ \mathrm{Sat}(N_3)\end{matrix}\right\}$	$\max\left\{\begin{matrix}\mathrm{Den}(N_2),\\ \mathrm{Den}(N_3)\end{matrix}\right\}$
$(N_2,N_3)\overset{\text{or}}{\longmapsto}N_1$	$\max\left\{\begin{matrix}\mathrm{Sat}(N_2),\\ \mathrm{Sat}(N_3)\end{matrix}\right\}$	$\min\left\{\begin{matrix}\mathrm{Den}(N_2),\\ \mathrm{Den}(N_3)\end{matrix}\right\}$
$N_2\overset{+S}{\mapsto}N_1$	$\min\left\{\begin{matrix}\mathrm{Sat}(N_2),\\ P\end{matrix}\right\}$	N
$N_2\overset{++S}{\longmapsto}N_1$	$\mathrm{Sat}(N_2)$	N
$N_2\overset{+D}{\mapsto}N_1$	N	$\min\left\{\begin{matrix}\mathrm{Den}(N_2),\\ P\end{matrix}\right\}$
$N_2\overset{++D}{\longmapsto}N_1$	N	$\mathrm{Den}(N_2)$
$N_2\overset{-S}{\mapsto}N_1$	N	$\min\left\{\begin{matrix}\mathrm{Sat}(N_2),\\ P\end{matrix}\right\}$
$N_2\overset{--S}{\longmapsto}N_1$	N	$\mathrm{Sat}(N_2)$
$N_2\overset{-D}{\mapsto}N_1$	$\min\left\{\begin{matrix}\mathrm{Den}(N_2),\\ P\end{matrix}\right\}$	N
$N_2\overset{--D}{\longmapsto}N_1$	$\mathrm{Den}(N_2)$	N

Each top goal is refined using AND/OR decomposition into subgoals. For example, G_1 is OR-decomposed into earn loan interest (G_2) or charge high fee for loan origination (G_3). Such decompositions model alternative ways of fulfilling a goal. Table 1 presents the rules used to propagate evidence through AND/OR relations (first two rows of the table). Thus, $\mathrm{Sat}(G_1)$ is assigned the maximum value among SAT values of all its subgoals (e.g., G_2 and G_3). Conversely, $\mathrm{Den}(G_1)$ is assigned 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_{15}). This 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 all leaf goals are tangible (i.e., there is an actor that can fulfill it).

The next step is to model relationships among goals using contribution relations. For instance, the goal receive electronic application (G_6) supports the goal verify loan application (G_{11}) (i.e., $G_6\overset{+s}{\mapsto}G_{11}$) because it promotes the possibility of doing automatic verification. As presented in Table 1, "+" and "−" relation propagates at most *partial* evidence from source nodes, while "++ " and "– –" propagate *full* evidence. Moreover, by means of these relations, we can model interrelations between the asset layer and the other layers (i.e., event and treatment layer). This allows us to model situations where a goal fulfillment increases/reduces the evidence of occurrence of an event, or a goal fulfillment supports/prevents the execution of countermeasures. For instance, the goal of having a loan application approved by clerk (G_{19}) will increase SAT of the event collusion between customer and clerk (E_{10}).

4.2 Event layer

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.

The notion of event used here is slightly different from *threat* in computer security literature [34] and *hazardous* condition in reliability engineering [30]. Those concepts are only defined as a potential circumstance that could cause harm or loss and do not include any notion of likelihood.[5]

Following the Probabilistic Risk Assessment (PRA) approach [8] and the ISO Guide 73 [22], the GR framework characterizes events with two properties: likelihood and severity. Likelihood is modeled as a property of an event (i.e., SAT and DEN), 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 a loan scheme 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 loan application. An event is a *risk* when it has negative impact (e.g., −, − −) to some elements of the asset layer; it is an *opportunity* when it produces positive impact (e.g., +, ++). This means an event may serve 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 loan interest (G_2). In such situations, it may not make sense to eliminate totally the risk, since the event also results in benefits. Rather, it may be better to reduce the risk likelihood to an acceptable level. Alternatively, analysts may introduce treatments that alleviate only the negative impact of the event to the asset layer.

The modeling of the event layer starts with event identification. There are different approaches for this, such as obstacle analysis [25], anti-goal [26], hazard analysis [24], misuse case [40], abuse case [33], taxonomy-base risk identification [14], or risk in finance [21]. Once events have been identified, they are decomposed into sub-events using similar decomposition relations as in the asset layer. This process continues until we reach leaf events that are easily observable. Throughout, it is important to ensure that all sub-events are distinct. To model dependency among events, one can use contribution relations, such as $E_{12}\overset{+s}{\longmapsto}E_{11}$ in the LOP scenario.

Table 2 Likelihood calculation based on evidence values

$\mathrm{Sat}(E) \wedge \mathrm{Den}(E) \longmapsto \lambda(E)$		
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

As already mentioned, an event is characterized by two properties: *likelihood* and *severity*. In our framework, we calculate the likelihood of an event ($\lambda(E)$) on the basis of the value of evidence that supports (i.e., SAT) and prevents (i.e., DEN) the occurrence of the event. The likelihood is defined qualitatively and can take the following values: *(L)ikely*, *(O)ccasional*, *(R)are*, and *(U)nlikely*, with intended meaning $L > O > R > U$. Table 2 defines the calculation rules of likelihood from SAT and DEN values. The table prescribes an event with *full* evidence of being satisfied and no evidence of denial as a *likely* event. Consequently, an event without any evidence of satisfaction results to be an *unlikely* event, independently of any denial evidence.

By severity we mean the effect of an event to the achievement of a goal asset. This definition is similar with the one in FMECA [43] or *impact* given in DDP [18]. Severity can take the following values:

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

This classification is encoded as the sign of an impact relation which connect the event layer with the asset layer. Impact relations introduce new evidence for the asset layer, and the value of new evidence depends on the likelihood of the event and the sign of impact relation (as defined in Table 3). The table specifies that an event propagates *full*

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

[5] Some security frameworks (e.g., CORAS [10]) or reliability engineering frameworks (e.g., PRA [8], FTA [42]) have incorporate the notion of likelihood in their modeling as part of the "risk" concept.

Table 3 Evidence propagation rules for impact relation

$\lambda(E)$ 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 4 Rewriting rules for alleviation relation

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

evidence of satisfaction to the asset layer if its likelihood is *likely* and connected by "++" impact relation (called as 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 as a risk). Moreover, through this representation an event may act as an opportunity and as a risk at the same time.

As indicated earlier, impact relations are special relations in $\langle \mathcal{N}, \mathcal{R}, \mathcal{I} \rangle$ because they model the impact of events on goals, but they also serve as target nodes for alleviation relations. An alleviation relation connects a treatment with an impact relation. This allows us to model a treatment as mitigation reduces severity (e.g., $- - \longmapsto -$). In this setting, we treat the impact relation as the target node of the alleviation relation. Details about alleviation relations are provided in the next subsection.

4.3 Treatment layer

We next focus on treatments/countermeasures/mitigations intended to mitigate risks. These can be analyzed using (AND/OR) decomposition and contribution relations, just like their asset cousins. A treatment may impact on a risk in two different ways: reducing its *likelihood* or attenuating its *severity*. To reduce the likelihood, a treatment 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 by applying rules of Table 2 it results in a less likely event.

More detail guidelines for eliciting treatments has been presented in [2]. Essentially, treatments are categorized into: removal/avoidance, prevention, attenuation, and retention depending on how they mitigate the risk in the event layer. Removal/avoidance measures tries to avoid selecting an alternative that has no relevant risks. Prevention aims to prevent or reduce the likelihood of any possibilities of the occurrence such a negative event. Differently, attenuation aims to reduce the severity of the risk, and retention does not mitigate either the likelihood or severity of the event. Accepting the risk per se or transferring the risk (e.g., insurance) can be considered as a retention measure.

To reduce severity of an event, we introduce the *alleviation* relation. This operates by reducing the *impact* sign to a lesser value. For instance (Fig. 2), 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 of E_2, but rather to reduce the severity of the risk E_2 toward G_6 (i.e., $T_1 \overset{-}{\mapsto} [E_2 \overset{--}{\longmapsto} G_6]$) by rewriting $E_2 \overset{--}{\longmapsto} G_6$ into $E_2 \overset{-}{\longmapsto} G_6$. The rules for alleviation relations are presented in Table 4. For simplicity, we only consider whether the treatment is selected or not and we do not take treatment evidence into account. Alleviation relations reduce only negative impact relations. The sign "$\emptyset$" indicates that there is no further impact between a treated event and a goal it was impacting.

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

Finally, we have defined a formal semantics for each relation in the GR framework in terms of formalizations rules that describe how evidence (`SAT` and `DEN`) is propagated from source to target node. A target node may have several incoming relations, and each relation may be introducing evidence to the target node. Accordingly, we need an evidence aggregation function (i.e., max).

4.4 Metamodel

In this section, we have introduced the basic concepts of the Tropos Goal-Risk for modeling and analyzing risks along the requirement analysis. All of those concepts are structured in terms of a meta-model depicted in Fig. 3.

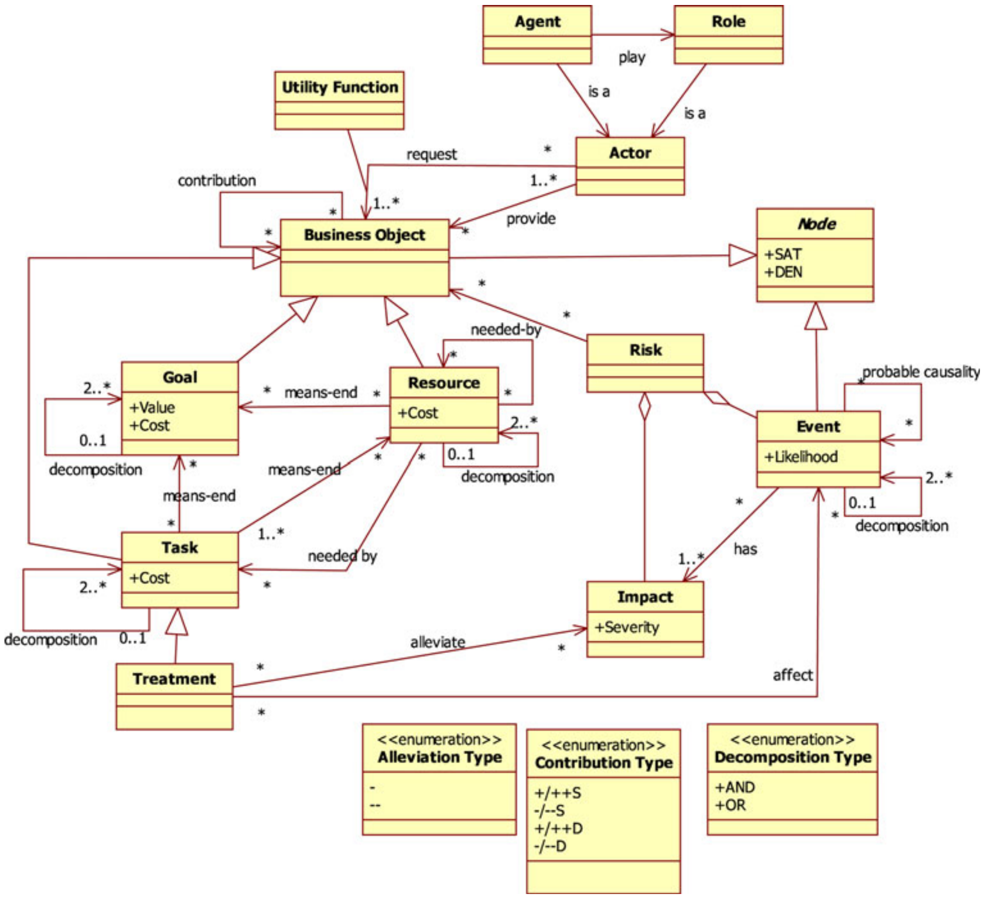


Fig. 3 Metamodel of the goal-risk modeling language

In Fig. 3, the model presents the basic constructs of the GR modeling language, such as: business objects (i.e., goals, tasks, and resources), events, and relations among concepts (e.g., decomposition, contribution, alleviation, means-end, needed-by). Moreover, impact is considered as a special relation because it can be a relation between an event and a business object, or it can act as a target node in the case of alleviation relation (as illustrated in $T_1 \xmapsto{-} [E_2 \xmapsto{--} G_6]$ at Fig. 2). In addition, those constructs have some properties characterizing them. For instance, goal, task, resource, and event have evidence values (i.e., SAT and DEN). Some relations (e.g., contribution, alleviation, decomposition) have several subtypes as indicated in Fig. 3.

5 Risk assessment process

In this section, we describe the methodological process and qualitative risk reasoning techniques used to analyze and evaluate alternatives in a GR model. Particularly, we focus on finding and evaluating all possible ways (called strategies) for satisfying top goals with an acceptable level of risk. In other words, given a GR model, each OR-decomposition introduces alternative modalities for top goal satisfaction, namely different sets of leaf 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, may introduce additional costs and further complications.

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 6–7)
3. assess the countermeasures to mitigate risks (line 9–16).

The process, shown in Fig. 4, takes a GR model as input, along with a desired satisfaction level for each root-level goal (*desired_labels*), acceptable risk values (*acc_risks*), evidence on goals as possible candidates for the final solution (*input_goals*), and finally evidence on events (*event_labels*). For instance, we might desire to have *full* evidence for satisfaction of goals G_4 and G_{10} (i.e., $\text{Sat}(G_4) = F$ and $\text{Sat}(G_{10}) = F$), while we do not care about goals G_1 and G_7 admitting *partial* evidence for their

Algorithm 1 Risk_Assessment_Process

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

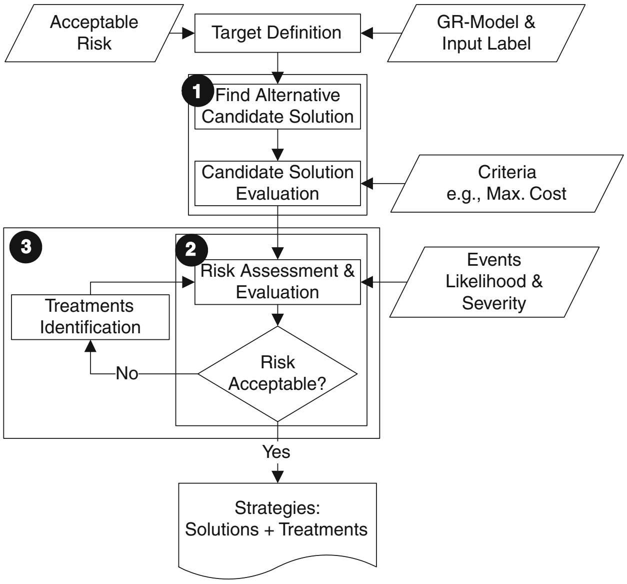

Acceptable Risk
Target Definition
GR-Model & Input Label
1
Find Alternative Candidate Solution
Candidate Solution Evaluation
Criteria e.g., Max. Cost
3
2
Risk Assessment & Evaluation
Events Likelihood & Severity
Treatments Identification
No
Risk Acceptable?
Yes
Strategies: Solutions + Treatments

Fig. 4 Risk assessment process flowchart

satisfaction (i.e., Sat(G_1) = P and Sat(G_7) = P). We may also require that there is no risk for G_1 and G_7 (i.e., Den(G_1) = N and Den(G_7) = N) while we allow *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 $\{G_2, G_3, G_5, G_6, G_8, G_9, G_{11}, G_{13}, G_{14}, G_{16}, G_{17}, G_{19}, G_{20}\}$.

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* amounts to the top–down reasoning mechanism proposed in [37], where a GR model is encoded into satisfiability formulas and then a SAT solver is used to enumerate all input-goals that can satisfy *desired_labels* for top goals. Here, the use of backward reasoning is limited to the asset layer (i.e., not considering the relations with the other two layers). For instance, in our example to achieve *desired_labels* we have 8 alternative solutions 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 alternative solutions, the analyst may select some on the basis of a criterion, such as maximum-cost for fulfilling all input goals. Suppose that the analyst decides to choose alternatives with a cost less than 30 (S4, S7, and S8 in our case—see Table 5).

Each *candidate_solution* is now evaluated against risk and possible countermeasures are introduced in case the risk is unacceptable (line 4–18). First, the analyst checks whether the *candidate_solution* (e.g., S4), together with risks in the event layer (*event_labels*), can still lead to desired levels of evidence for top goals. To accomplish this, we use *Forward_Reasoning* (adapted from [19]) which propagates evidence labels throughout the GR model. Later, *Satisfy* compares final evidence labels for top goals with those 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/greater than the desired values (*desired_labels*), then the *candidate_solution* is added directly to the *solution* and its cost is calculated (line 7). Otherwise, countermeasures must be introduced in the *candidate_solution* (line 10–16). The adaptation of *Forward_Reasoning* is detailed at the end of this section.

In order to define countermeasures, the analyst calculates the maximum DEN values of input goals (*max_labels*) that

Algorithm 2 Find_Treatments

```
Require: gr_model $\langle\mathcal{N},\mathcal{R},\mathcal{I}\rangle$, label_array current, label_array max
1: array rel_events, rel_impact, rel_treatments
2: for all $c_i \in current \wedge is\ Goal(c_i)$ do {find all events that threaten the asset layer}
3:     if $c_i.den > max_i.den$ then
4:         tmp_events ← $related(\langle\mathcal{N},\mathcal{R},\mathcal{I}\rangle, c_i, 'impact')$
           {find all related events to the goal $c_i$}
5:         $add(rel\_events,\ tmp\_events)$
6:         tmp_impact ← $R_k \in \mathcal{I}$ s.t. $source(R_k) = e_j \wedge target(R_k) = c_i$
           {identify severity/impact relation of the event}
7:         $add(rel\_impacts,\ tmp\_impact)$
8:     end if
9: end for
10: for all $e_i \in rel\_events$ do {find all possible treatments to reduce the likelihood of the events}
11:     tmp ← $related(\langle\mathcal{N},\mathcal{R},\mathcal{I}\rangle, e_i, 'negative-contribution')$
        {treatments for reducing likelihood}
12:     $add(rel\_treatments,\ tmp)$
13: end for
14: for all $i_i \in rel\_impact$ do {find all possible treatments to alleviate the severity of the events}
15:     for all $R_j \in \langle\mathcal{N},\mathcal{R},\mathcal{I}\rangle$ s.t. $target(R_j) = i_i$ do {identify all alleviation relations}
16:         $add(rel\_treatments,\ source(R_j))$
17:     end for
18: end for
19: return treatments
```

Algorithm 3 Forward_Reasoning

```
Require: gr_model $\langle\mathcal{N},\mathcal{R},\mathcal{I}\rangle$, label_array initial
1: label_array current, old
2: current ← initial
3: i ← 0
4: repeat
5:     while $old \neq current$ do
6:         old ← current
7:         for all $N_j \in \mathcal{N}$ do
8:             current$_j$ ← $Update\_Label(i, \langle\mathcal{N},\mathcal{R},\mathcal{I}\rangle, old, initial)$
9:         end for
10:    end while
11:    if i = 0 then
12:        $Apply\_Alleviation(\langle\mathcal{N},\mathcal{R},\mathcal{I}\rangle, current)$
13:        old ← nil
14:    end if
15:    $i$ ++
16: until i = 2
17: return current
```

produce acceptable DEN values for top goals. In other words, we need to find a set of countermeasures able to mitigate risk, so that we end up with acceptable risk levels (*acc_risks*). We calculate *max_labels* values using *Backward_Reasoning* (line 9) and specify the goals in the *candidate_solution* as input goals and *acc_risks* as a constraint for the DEN value of input goals.By having two sets of values (i.e., *cur_labels* and *max_labels*), we can find treatments that mitigate risks and at the same time bring up evidence of top goals to acceptable levels (e.g., *max_labels*).

To identify treatments, we propose *Find_Treatments* (Algorithm 2) which enumerates all possible sets of treatments that can mitigate risks to acceptable levels. By comparing DEN in the *max_labels* and the *cur_labels*, one can identify which goals are overvalued. For instance, given S4 (Table 5) the goal G_6 in *max_labels* is defined as Den(G_6) = P, whereas in *cur_labels* it is Den(G_6) = F (see Table 6 column Event-Out). Next, we need to define related events that may cause this level of risk[6]. In our case, the relevant events for G_6 are E_2, E_3, and E_4. Moreover, we need to identify the severity of those events. Once we have identified the *rel_events*, we can find the treatments (*rel_treatments*) that can mitigate these events (line 10–18). As discussed earlier, mitigations may reduce likelihood and/or severity of a risk. In the case of E_2, E_3, and E_4, the treatments that reduce the severity are T_2 and T_3, while T_4 and T_5 reduce likelihood.

However, it could be the case that a treatment in *rel_treatments* has an overlapped effect in mitigating risks with other treatments. Thus, we evaluate each subset of *rel_treatments* to check whether it is sufficient 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*. In this way, we are able to seek the strategy which is most cost-effective. Moreover, through this evaluation, analysts can estimate the side effect of additional treatments to the asset layer besides their effect to the event layer. If the *desired_labels* and *acc_risks* are achieved then the treatments and the input goals can be added to the *solution* and the cost is calculated (Algorithm 1 line 14).

Forward_Reasoning (Algorithm 3), essentially, is the adaptation of the one proposed in [19]. 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 Table 1 for decomposition and contribution relations (line 6–7) and Table 3 for impact relation (line 3–4). Based on *current* evidence values, final

[6] *Related* is meant to enumerate all the nodes that can be reachable from a given node in $\langle\mathcal{N},\mathcal{R},\mathcal{I}\rangle$ through a particular type of relation.

Algorithm 4 Update_Label

```
Require: int i, gr_model $\langle \mathcal{N}, \mathcal{R}, \mathcal{I} \rangle$, label_array old
1: for all $R_j \in \mathcal{R} \cup \mathcal{I}$ s.t. $target(R_j) = N_i \wedge \neg is\_Alleviation(Rj)$ do
2:     if $is\_Impact(R_j)$ then
3:         $sat_{ij} = Apply\_Imp\_Sat(N_i, R_j, old)$
4:         $den_{ij} = Apply\_Imp\_Den(N_i, R_j, old)$
5:     else {decomposition and contribution relations}
6:         $sat_{ij} = Apply\_Rules\_Sat(N_i, R_j, old)$
7:         $den_{ij} = Apply\_Rules\_Den(N_i, R_j, old)$
8:     end if
9: end for
10: return $\{max(max\_array(sat_{ij}), initial[i].sat),$
      $max(max\_array(den_{ij}), initial[i].den)\}$
```

Algorithm 5 Apply_Alleviation

```
Require: gr_model $\langle \mathcal{N}, \mathcal{R}, \mathcal{I} \rangle$, label_array current
1: for all $I_k \in \mathcal{I}$ do
2:     for all $R_l \in \mathcal{R}$ s.t. $target(R_l) = I_k \wedge is\_Alleviation(R_l)$ do
3:         $I_k \leftarrow Update\_Sign(R_l, current)$
4:     end for
5: end for
```

evidence values of adopted treatments are estimated. Thus, using *Apply_Alleviation* (Algorithm 5) we rewrite the sign of impact relations that are alleviated by treatments—the rules (Table 4) are encoded in the line 3 of the algorithm. Afterward, the rewritten GR model $\langle \mathcal{N}, \mathcal{R}, \mathcal{I} \rangle$ is again evaluated in the second loop. Here, the evidence values of all nodes in the GR model are final, and consequently the risk levels of top goals are estimated (i.e., DEN of top goals).

6 Framework validation

The Tropos Goal-Risk Framework has been used to model and analyze several case study such as the London Ambulance System (LAS) [2], the partial airspace delegation in Air Traffic Management (ATM) [4], and the Intra-Manufacturing Small-Medium Enterprises (SMEs) [7]. An important objective for all case studies was to evaluate the expressiveness of the modeling framework and validate the formal analysis that it supports. Moreover, in the ATM scenario, we evaluated the usability of the framework by software practitioners (i.e., our industrial partners) [5]. These case studies were developed by gathering information from domain experts using a questionnaire[7], and at the ATM case study we conduct a focus group discussion [39] to evaluate and gather feedback about the GR framework. Moreover, only at ATM and SMEs case studies where we perform not only risk modeling and analysis. In these case studies, we confirm the outcomes of automated reasoning with the domain experts. From these case studies, we have learned that (1) the GR framework is sufficient to capture basic concepts relevant to risk and requirement analysis, (2) it is hard to validate the inputs of assessments, since the system has not been implemented, (3) precision of the assessment method at this phase is less important since domain experts cannot feel the difference the risk level smaller than 2 decimal digits, and at last (4) the requirement schema has proven to be effective in developing the GR model rather than teaching them to use the tool and expect they will model their problem using the modeling framework.

In this section, we include an experimental evaluation of the scalability of the proposed automated reasoning technique using the LOP scenario. To help analysts in modeling, we have developed a tool that is an extension of the Goal Reasoning Tool (GR-Tool)[8] developed within the Tropos project and as one feature of the SI* Tool[9] developed within the SERENITY project. Basically, the tool (Fig. 5) 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 ran a number of experiments with the *loan origination process* scenario which is a simplification of Serenity e-Business scenario [4]. In the following, we discuss the scenario for our experiment. Suppose we are interested in *partial* evidence for the satisfaction of top-goals earn more income (G_1) and ensure repayment of loan (G_7), while we insist on *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 $\text{Den}(G_4) = P$, -$\text{Den}(G_{10}) = P$, $\text{Den}(G_1) = N$, and $\text{Den}(G_7) = N$. Given these inputs, the set of possible solutions is reported in Table 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, suppose we decide to focus on ones with a cost less than 30. These are S4, S6, S7, and S8. Particularly, let us consider S4 with the initial assignment for input goals as reported in Table 6 (column "Goal-In"). This assignment satisfies the desired values for top-goals (column "Goal-Out"). Now, if we introduce the assignment to

[7] The questionnaire is called requirement collection schema and available at http://fmsweng.science.unitn.it/~sistar/

[8] http://www.disi.unitn.it/~yudis/tools/GrTool.jar

[9] http://www.sesa.dit.unitn.it/sistar_tool; http://fmsweng.science.unitn.it/~sistar/

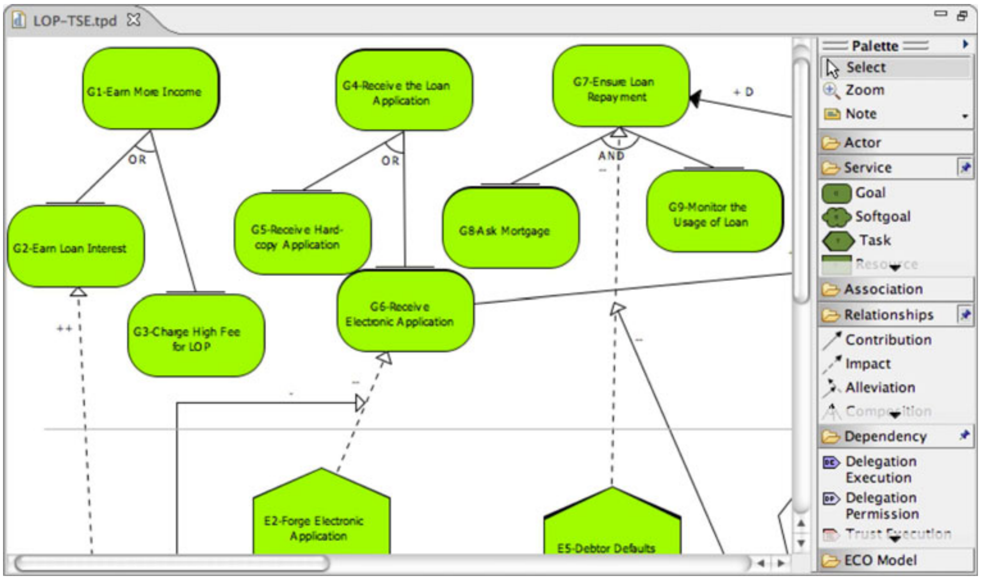


Fig. 5 Goal risk tool

Table 5 Cost of alternative solutions

Input goal	Cost	S1	S2	S3	S4	S5	S6	S7	S8
G2-earn loan interest	3	X		X		X		X	
G3-charge high fee for LOP	2		X		X		X		X
G5-receive hard-copy app.	5	X	X			X	X		
G6-receive electronic app.	3			X	X			X	X
G8-ask mortgage	2	X	X	X	X	X	X	X	X
G9-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

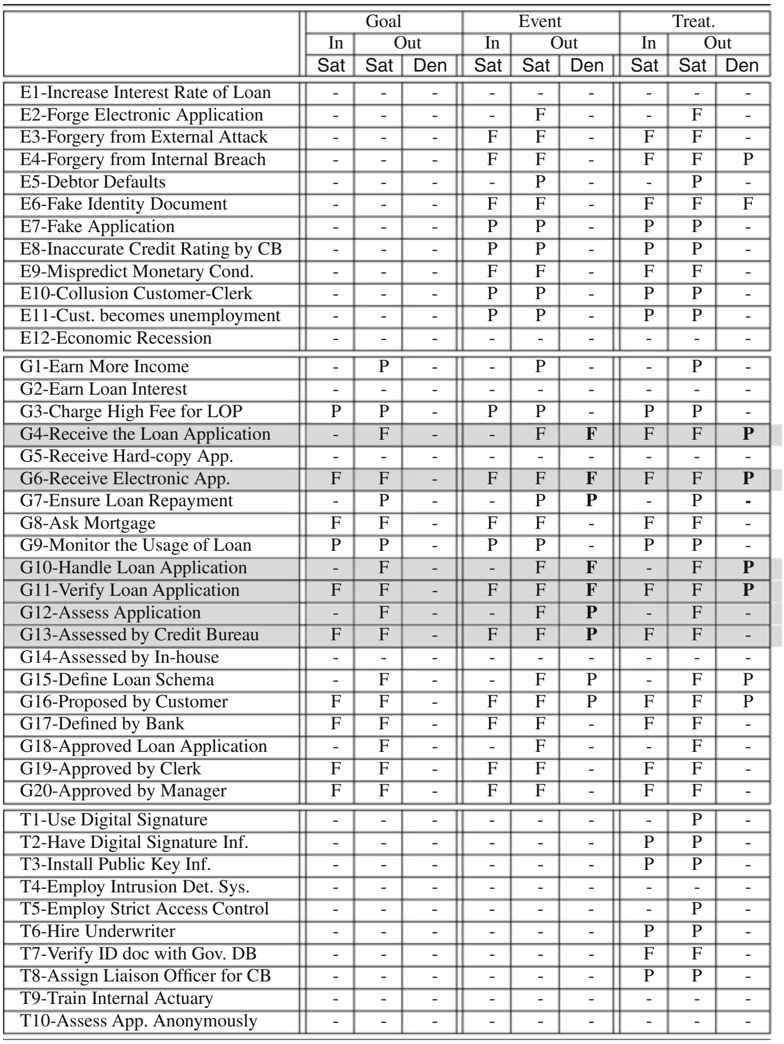
events reported in column “Event-In”, the desired values for top goals are no longer satisfied (“Event-Out”). Readers might compare SAT and DEN values in the cell at grey rows. For instance, G_4 has *full* DEN evidence, while the acceptable value was at most *partial*. This happens since goal receive 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 Table 7. As reported in column “Treat-Out” in Table 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.

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 Table 7.

Figure 6 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.

Moreover, we have also tested the performance of the implementation of the forward reasoning algorithm with several much bigger test cases. Actually, these cases are generated using on a GR model from a real case study as the basis. A big model comes from joining several GR models with random GR relations. We conducted the experiment using Java(TM) Runtime Environment (1.4.2) and a machine with Intel(R) Xeon(TM) CPU 2.40 GHz and 1.5GB RAM. Though theoretically the algorithm has linear complexity, in the experiment it demonstrated different behaviour. The chart (in Fig. 7) shows that execution time grows exponentially with the size of the GR model. Moreover, the increase of relations adversely affects the performance of the reasoner, more than the increase of the number of nodes. This phenomenon is caused by background

Table 6 `SAT-DEN` values of S4-alternative and C1-treatments



	Goal			Event			Treat.		
	In	Out		In	Out		In	Out	
	Sat	Sat	Den	Sat	Sat	Den	Sat	Sat	Den
E1-Increase Interest Rate of Loan	-	-	-	-	-	-	-	-	-
E2-Forge Electronic Application	-	-	-	-	F	-	-	F	-
E3-Forgery from External Attack	-	-	-	F	F	-	F	F	-
E4-Forgery from Internal Breach	-	-	-	F	F	-	F	F	P
E5-Debtor Defaults	-	-	-	-	P	-	-	P	-
E6-Fake Identity Document	-	-	-	F	F	-	F	F	F
E7-Fake Application	-	-	-	P	P	-	P	P	-
E8-Inaccurate Credit Rating by CB	-	-	-	P	P	-	P	P	-
E9-Mispredict Monetary Cond.	-	-	-	F	F	-	F	F	-
E10-Collusion Customer-Clerk	-	-	-	P	P	-	P	P	-
E11-Cust. becomes unemployment	-	-	-	P	P	-	P	P	-
E12-Economic Recession	-	-	-	-	-	-	-	-	-
G1-Earn More Income	-	P	-	-	P	-	-	P	-
G2-Earn Loan Interest	-	-	-	-	-	-	-	-	-
G3-Charge High Fee for LOP	P	P	-	P	P	-	P	P	-
G4-Receive the Loan Application	-	F	-	-	F	**F**	F	F	**P**
G5-Receive Hard-copy App.	-	-	-	-	-	-	-	-	-
G6-Receive Electronic App.	F	F	-	F	F	**F**	F	F	**P**
G7-Ensure Loan Repayment	-	P	-	-	P	**P**	-	P	**-**
G8-Ask Mortgage	F	F	-	F	F	-	F	F	-
G9-Monitor the 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-Approved 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	-
T1-Use Digital Signature	-	-	-	-	-	-	-	P	-
T2-Have Digital Signature Inf.	-	-	-	-	-	-	P	P	-
T3-Install Public Key Inf.	-	-	-	-	-	-	P	P	-
T4-Employ Intrusion Det. Sys.	-	-	-	-	-	-	-	-	-
T5-Employ Strict Access Control	-	-	-	-	-	-	-	P	-
T6-Hire Underwriter	-	-	-	-	-	-	P	P	-
T7-Verify ID doc with Gov. DB	-	-	-	-	-	-	F	F	-
T8-Assign Liaison Officer for CB	-	-	-	-	-	-	P	P	-
T9-Train Internal Actuary	-	-	-	-	-	-	-	-	-
T10-Assess App. Anonymously	-	-	-	-	-	-	-	-	-

memory allocation processes that execute during the reasoning process. In a small scenario (i.e., below 50 nodes/constructs in a GR model), the reasoner can easily obtain the memory needed, but in a bigger scenario this allocation takes more time. Additionally, the performance of *Backward_Reasoning* is closely dependent on the performance of the SAT solver that is used. Finally, the performance of the risk assessment process depends on the number of treatments that may be adopted to mitigate risks, as discussed in the previous sections. Notice that the assessment process still requires human intervention in selecting the alternatives (i.e., *Select_Can_Solution*) to be analyzed further. Actually, this step may be skipped. However, this step significantly reduces the space of possible alternatives, and consequently results in better performance for the overall risk assessment process. The final decision of choosing among possible strategies is left to the analysts.

7 Pros and cons of the goal-risk framework

Starting with pros, we had positive experiences in communicating GR models to analysts and domain experts. This is an important strength for any requirements analysis technique because it empowers domain experts to understand and critique proposed models [4]. Moreover, the learning process for experts to understand and use a GR model takes relatively short period (approximately 2–3 months). This is partly due to the familiarity experts had with the concepts of goal and task. In [5] we report on the usage of the GR framework as part of the SI* modeling framework in identifying security and dependability patterns, and in [7] the GR framework was used to assess the possible business solutions in a manufacturing SME.

The GR framework supports risk analysis during the very early phases of software development. Consequently,

Table 7 Cost of possible treatments

Treatment	Cost	S4				S6	S7/S8			
		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

it reduces the risk of requirements revision, and consequently the cost of development. This framework has been tried for analyzing requirements and risks at various critical information systems (e.g., business, safety, and mission critical) [41]. However, we believe the framework can be used at other domain application, but surely it needs some adjustments. By having an asset layer consisting of stakeholder goals, one can focus of those goals. Other frameworks focus on risks for processes and resources that are means for achieving stakeholder goals. Note that an event may not be a risk for these processes and resources, but may obstruct the corresponding goal nonetheless. For instance, the goal `increase sales` may be achieved by `advertise products` and `discount products`. Let us assume that both means are undisrupted. However, the event `new competition` may disrupt the goal itself, but not the means of the goal. Indeed, this paper has focused on goals as assets, rather than the means for achieving these goals. However, in other works [3, 6] we have also studied risks on the means to achieve goals by introducing the notion of process to achieve a goal and artifacts that are required by such processes. The GR framework can also be used to assess risks at multi-actors setting where the impact of risk can be propagated among actors following the dependency among them [7].

Many risk modeling frameworks only consider risks, i.e., events that can have negative impact on the mission of the system-to-be. However, events may also have positive impact. Within the GR framework, analysts can capture such situations, and consequently perform trade-off analysis to find mitigations that result in acceptable risk levels without preempting any opportunities. The three layers of the GR model facilitate extensions to accommodate features of other frameworks, such as: FTA, FMECA, Attack-Graph, and Markov-Model. However, such extensions may also require extensions to the range of evidence values (e.g., 5 values instead only 3). This flexibility is one of added advantage with respect to other related works (e.g., KAOS, DDP, CORAS). In comparison with KAOS, this framework allows analysts to perform qualitative and quantitative assessment though KAOS provides richer formal semantics using Linear Temporal Logic. Moreover, in comparison with DDP and CORAS the GR framework is more expressive in capturing stakeholders' intentions. At last, the GR framework is the only framework that deals with risk and opportunity, since some risks appear because the stakeholders decide to pursue an opportunity. With this feature, one can perform trade-off analysis to decide whether one opportunity is worth to pursue or not.

Besides those pros, there are weaknesses in the proposed framework. First, the notion of evidence (`SAT` and `DEN`) is not as well understood as *probability*, which is most often

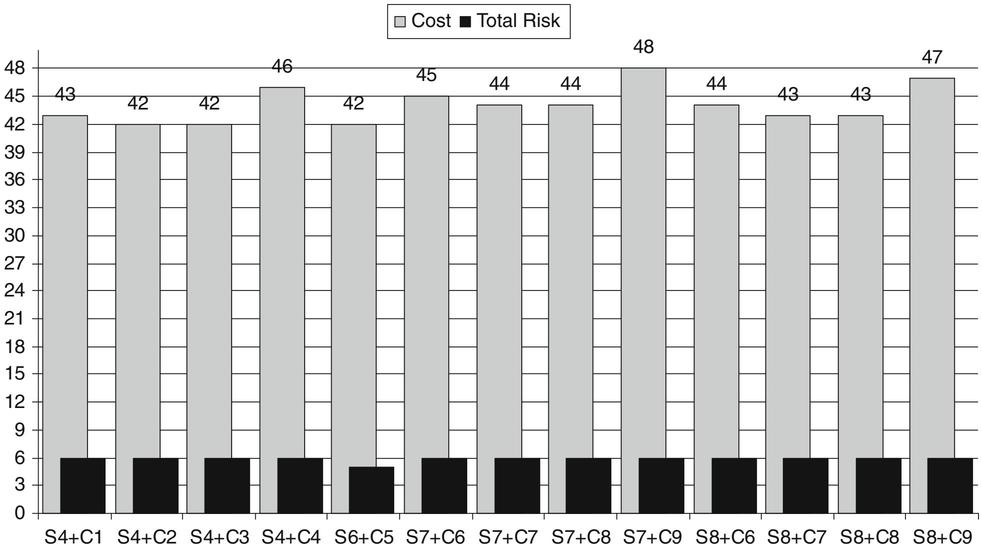


Fig. 6 Comparison total risk and total cost among all candidate alternatives and their treatments

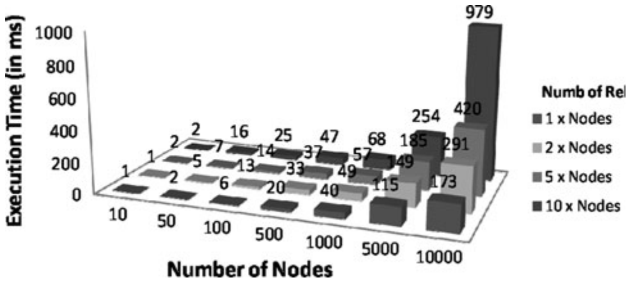


Fig. 7 Execution time of forward reasoning algorithm

used for risk analysis. In response to this, we have extended the framework to be able to accept both inputs (e.g., evidence and probability) [7]. However, we note that our proposed risk analysis is not meant to assess precisely risk levels. Rather, it is meant to give designers guidance on how to produce designs that minimize risks to the fulfillment of stakeholder needs. For projects where budgets are limited, our risk analysis can point to high risks so that they can be given priority. Of course, this analysis depends on value judgments made by the analysts in determining levels of likelihood and severity. These judgments are typically determined collectively by a group of analysts. For a more formal approach, one may adopt techniques such as the Delphi method [27] where an iterative process results in stable judgments for all participants.

8 Conclusions and future work

In this article, we have presented a framework for modeling and analyzing risk during the requirements engineering process. Our framework adopts and extends the Tropos goal modeling framework and proposes qualitative reasoning algorithms to analyze risk during the process of evaluating and selecting among alternative designs. Moreover, this work extends the DDP framework by introducing some relations that result in a more expressive modeling framework. These relations capture correlations among events and the distinction between treatments that aim at reducing the likelihood vs attenuating the impact of an event.

This modeling framework is equipped with a methodological process and two CASE tools (e.g., the GR tool and SI* Tool) that support analysts in capturing, depicting, and analyzing user requirements and their related risks. Moreover, we have also validated this framework in terms of: (1) usability using several case studies and (2) performance with bigger models that are generated using the real case study model as a basis.

For future work, we would like to extend our analysis to larger sets of SAT and DEN values, also to study quantitative reasoning mechanisms where evidence is expressed in term of [0, 1] as in the Dempster-Shaffer theory.

Acknowledgments This work has been partly supported by the projects EU-SERENITY, EU-MASTER, FIRB-ASTRO, PAT-MOSTRO, PAT-STAMPS, and PRIN-MENSA.

References

1. Anton AI (1996) Goal-based requirements analysis. In: Proceedings of the 2nd IEEE international conference on requirements engineering (ICRE'96), IEEE Computer Society Press, Washington, DC, USA, p 136
2. Asnar Y, Giorgini P (2006) Modelling risk and identifying countermeasures in organizations. In: Proceedings of the 1st international workshop on critical information infrastructures security, Springer-Verlag, Lecture Notes in Computer Science, vol 4347, pp 55–66
3. Asnar Y, Giorgini P (2008) Analyzing business continuity through a multi-layers modell. In: Proceedings of 6th international conference on business process management
4. Asnar Y, Bonato R, Bryl V, Campagna L, Dolinar K, Giorgini P, Holtmanns S, Klobucar T, Lanzi P, Latanicki J, Massacci F, Meduri V, Porekar J, Riccucci C, Saidane A, Seguran M, Yautsiukhin A, Zannone N (2006) Security and privacy requirements at organizational level. Project deliverable A1.D2.1, SERENITY consortium, EU-IST-IP 6th framework programme—SERENITY 27587
5. Asnar Y, Bonato R, Giorgini P, Massacci F, Meduri V, Riccucci C, Saidane A (2007a) Secure and dependable patterns in organizations: an empirical approach. In: Proceedings of the 15th IEEE international requirements engineering conference, IEEE Computer Society Press, Oakland, CA
6. Asnar Y, Giorgini P, Massacci F, Zannone N (2007b) From trust to dependability through risk analysis. In: Proceedings of the second international conference on availability, reliability and security, IEEE Press, New York
7. Asnar Y, Moretti R, Sebastianis M, Zannone N (2008) Risk as dependability metrics for the evaluation of business solutions: a model-driven approach. In: Proceedings of the third international conference on availability, reliability and security
8. Bedford T, Cooke R (2001) Probabilistic risk analysis: foundations and methods. Cambridge University Press, Cambridge
9. Boehm BW (1991) Software risk management: principles and practices. IEEE Softw 8(1):32–41. doi:10.1109/52.62930
10. den Braber F, Dimitrakos T, Gran BA, Lund MS, Stølen K, Aagedal JØ (2003) The CORAS methodology: model-based risk assessment using UML and UP. In: UML and the Unified Process, Idea Group Publishing, Hershey, pp 332–357
11. Bresciani P, Perini A, Giorgini P, Giunchiglia F, Mylopoulos J (2004) Tropos: an agent-oriented software development methodology. J Auton Agent Multi Agent Syst 8(3):203–236. doi:10.1023/B:AGNT.0000018806.20944.ef
12. Butler S, Fischbeck P (2001) Multi-attribute risk assessment. Tech. Rep. CMU-CS-01-169, Carnegie Mellon University
13. Butler SA (2002) Security attribute evaluation method: a cost-benefit approach. In: Proceedings of the 24th international conference on software engineering, ACM Press, New York, NY, USA, pp 232–240. doi:10.1145/581339.581370
14. Carr MJ, Konda SL, Monarch I, Ulrich FC, Walker CF (1993) Taxonomy-based risk identification. Tech. Rep. CMU/SEI-93-TR-6, Software Engineering Institute, Carnegie Mellon University

15. Chung LK, Nixon BA, Yu E, Mylopoulos J (2000) Non-functional requirements in software engineering. Kluwer Academic Publishers, Dordrecht
16. Cornford SL, Feather MS, Heron VA, Jenkins JS (2006) Fusing quantitative requirements analysis with model-based systems engineering. In: Proceedings of the 14th IEEE international requirements engineering conference, IEEE Computer Society Press, Los Alamitos, CA, USA, pp 279–284, doi:10.1109/RE.2006.24
17. Dardenne A, van Lamsweerde A, Fickas S (1993) Goal-directed requirements acquisition. Sci Comput Program 20(1–2):3–50
18. Feather MS (2004) 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, IEEE Computer Society Press, Silver Spring, MD, pp 391–402
19. Giorgini P, Mylopoulos J, Nicchiarelli E, Sebastiani R (2003) Formal reasoning techniques for goal models. J Data Semantics 1(1):1–20
20. Helmer G, Wong J, Slagell M, Honavar V, Miller L, Lutz R (2002) A software fault tree approach to requirements analysis of an intrusion detection system. Requir Eng J 7(4):207–220
21. Holton GA (2004) Defining risk. Financ Anal J 60(6):19–25
22. ISO/IEC (2002) Risk management-vocabulary-guidelines for use in standards. ISO/IEC Guide 73
23. Jürjens J (2001) Towards secure systems development with UMLsec. In: Proceedings of the 4th international conference on fundamental approaches to software engineering, Springer, New York, pp 187–200
24. Kletz TA (1997) HAZOP—past and future. Reliab Eng Syst Saf 55(3):263–266
25. van Lamsweerde A, Letier E (2000) Handling obstacles in goal-oriented requirements engineering. IEEE Trans Softw Eng 26(10):978–1005. doi:10.1109/32.879820
26. van Lamsweerde A, Brohez S, Landtsheer RD, Janssens D (2003) 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
27. Linstone H, Turoff M (1975) The delphi method: techniques and applications. Addison-Wesley Pub. Co, Reading, MA
28. Liu L, Yu ESK, Mylopoulos J (2003) Security and privacy requirements analysis within a social setting. In: Proceedings of the 11th IEEE international requirements engineering conference, pp 151–161
29. Lodderstedt T, Basin D, Doser J (2002) 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, Springer-Verlag, Lecture Notes in Computer Science, vol 2460, pp 426–441
30. Lutz RR, Woodhouse RM (1997) Requirements analysis using forward and backward search. Ann Softw Eng 3(1):459–475
31. Mayer N, Rifaut A, Dubois E (2005) Towards a risk-based security requirements engineering framework. In: Proceedings of the 11th international workshop on requirements engineering: foundation for software quality
32. Mayer N, Dobuis E, Rifaut A (2007) Requirements engineering for improving business/IT alignment in security risk management methods. In: Proceedings of the 3rd international conference interoperability for enterprise software and applications
33. McDermott J, Fox C (1999) Using abuse case models for security requirements analysis. In: Proceedings of 15th annual computer security applications conference, Phoenix, AZ, USA, pp 55–64. doi:10.1109/CSAC.1999.816013
34. Pfleeger CP, Pfleeger SL (2006) Security in computing, 4th edn. Prentice-Hall, Englewood, Cliffs, NJ
35. Roy GG, Woodings TL (2000) A framework for risk analysis in software engineering. In: Proceedings of the seventh Asia-Pacific software engineering conference (APSEC '00), IEEE Computer Society Press, Washington, DC, USA, p 441
36. Schneier B (1999) Attack trees: modeling security threats. Dr Dobbï J 12(24):21–29
37. Sebastiani R, Giorgini P, Mylopoulos J (2004) Simple and minimum-cost satisfiability for goal models. In: Proceedings of the 16th conference on advanced information systems engineering, Springer-Verlag Heidelberg, Lecture Notes in Computer Science, vol 3084, pp 20–33. doi:10.1007/b98058
38. Shafer G (1976) A mathematical theory of evidence. Princeton University Press, Princeton, NJ
39. Shull F, Singer J, Sjøberg DIK (2007) Guide to advanced empirical software engineering. Springer, New York
40. Sindre G, Opdahl AL (2005) Eliciting security requirements with misuse cases. Requir Eng J 10(1):34–44. doi:10.1007/s00766-004-0194-4
41. Sommerville I (2004) Software engineering, 7th edn. Addison Wesley, Reading, MA
42. Stamatelatos M, Vesely W, Dugan J, Fragola J, Minarick J, Railsback J (2002) Fault tree handbook with aerospace applications. NASA
43. US-DoD (1980) Military standard, procedures for performing a failure mode, effects, and critical analysis. MIL-STD-1629A
44. Yu E (1995) Modelling strategic relationships for process engineering. PhD thesis, University of Toronto, Department of Computer Science