Multi-Objective Risk Analysis with Goal Models

Fatma Başak Aydemir, Paolo Giorgini, John Mylopoulos
University of Trento
Trento, Italy
email: {fatmabasak.aydemir, paolo.giorgini, john.mylopoulos}@unitn.it

***Abstract*—Risks of software projects are often ignored and risk analysis is left for later stages of project life-cycle, resulting in serious financial losses. This paper proposes a goal-oriented risk analysis framework that includes inter-dependencies among treatments and risks in terms of likelihood and generate optimal solutions with respect to multiple objectives such as goal rewards, treatment costs, or risk factor. The Loan Origination Process illustrates our approach and a detailed analysis of the visual notation is provided.**

***Index Terms*—risk analysis, goal modeling, satisfiability modulo theories, optimization modulo theories, multi-objective optimization**

I. Introduction

Risk analysis should be an integral part of any development project for complex socio-technical systems intended to operate within an uncertain environment. Yet studies show that risks are highly underestimated in IT projects leading repeatedly to disastrous financial losses, delays, or total failures [1]. Risk analysis involves risk assessment and management, the activities for the first comprise risk identification and evaluation whereas the latter requires the selection of treatments to prevent or at least ameliorate to the implications of risks [2].

Existing approaches to risks identify situations that may lead to risks, assess the impact of risks on the system, and introduce treatments to mitigate the impact. The treatments may refine existing design, or even change the initial requirements [3]. However changing the requirements in later stages of software development adds additional cost, thus integrating risk analysis with requirements analysis has financial benefits in addition to leading to more robust designs [4].

Goal models have been used to capture and hierarchically structure requirements for sociotechnical systems. Higher level goals capture stakeholder needs whereas lower level goals capture strategies for fulfilling higher-level ones. The structure of goal models supports systematic analysis techniques to determine solutions to root-level goals [5]. Goal models can also be used to reason about security and trust for a system-to-be [6]–[8].

Considering the advantages of goal-oriented requirements engineering and integrating risk analysis with requirements analysis, Asnar et al. [9] proposes a goal-risk analysis framework. Goal-risk models capture stakeholder requirements, risks, and treatments, and support their analysis to find the optimal solutions with respect to cost. This approach focuses on avoiding risk while minimizing cost.

Alternative ways of managing risk include taking risks in exchange of possible greater benefits, accepting risk, and preparing a contingency plan. For the former, the solutions are optimized with respect to a utility function of stakeholder goals, for the latter, treatments to impact of risks should be modeled and analyzed. Cost is a factor that cannot be ignored for both cases. So a thorough risk analysis requires multi-objective optimization and various types of analysis to identify alternative solutions. In many cases, the analysts consider trade–offs between cost, risk aversion, utilities, and so on.

We adopt Asnar's framework, and extend it with constrained goal model [10] in order to support multiple types of risks analyses for different risk management strategies. Our extension exploits Satisfiability Modulo Theories/Optimization Modulo Theories (SMT/OMT) solvers to support efficient reasoning and discovery of solutions for risk analysis problems. We define multiple objective functions such as maximum goal reward, minimum risk factor, maximum risk prevention, minimum cost, and minimum damage on risk models and discover globally optimal solutions with respect to multiple objectives.

The rest of the paper is structured as follows. Section II presents the research baseline of this work. The modelling framework is presented in Section III, and Section IV explains SMT/OMT-based risk analysis in detail. Section V compares the contribution to related work in the literature, while Section VI concludes.

II. Research Baseline

Goal and Risk Models. Goal models have been widely used to capture and analyze requirements. Goal models allow a hierarchical representation for requirements, where high–level goals can be AND- or OR- refined into sub-goals. Besides goals and refinements, softgoals and tasks are included as first class concepts in many goal–oriented frameworks (NFR [11], KAOS [12], i* [13], and Tropos [14]). Van Lamsweerde explicitly models the goals of an attacker, an agent that creates security risks for the systems goals [15]. Anti-goal modeling starts by negating some of the goals of the system that are vulnerable to an attack. By asking 'why?' question more higher–level goals of the attacker are discovered. Then, the high–level goals are AND- and OR- refined to identify the low–level goals

978-1-4799-8710-8/16/$31.00 ©2016 IEEE

that are more concrete. The generated goal model is called anti-model. We adopt the notion of anti-goals to model the goals behind the intentional risks, which are created as a result of the actions of another system.

Asnar et al. [9] provides label propagation algorithms to asses risk in goal models. The goal models have asset, event, and treatment layers. The asset layer includes the goals of systems. The event layer includes events related to the system goals. Lastly, the treatment layer is the layer for countermeasures and treatments against risks that are introduced in the event layer and affect the asset layer. We propose a goal-oriented risk analysis technique where we borrow concepts such as goal, risk, impact, and treatment from [9] and enrich the metamodel with interdependencies and additional concepts. Asnar et al. discover optimal solution with respect to a single objective, which is cost, using label propagation algorithms. This paper uses SMT/OMT reasoning to discover optimal solution with respect to multiple objectives such as minimized likelihood, impact, treatment cost, and maximized gal reward.

SMT/OMT Reasoning. A satisfiability modulo theory (SMT) problem is the problem of assigning values to a set of variables of a set of first order logic formulas. If there is an assignment for variables that satisfies all formulas, the problem is satisfiable (SAT), otherwise it is unsatisfiabile (UNSAT). Optimization Modulo Theories (OMT) problems is a combination of SMT and optimization problems. In this case, the aim is find an assignment that minimizes (or maximizes) a given objective function while satisfying the formulas [16]. OptiMathSat is an SMT/OMT solver developed by Sebastiani and Trentin [17]. Nguyen et al. propose constrained goal models by combining SMT/OMT reasoning and goal models to find optimal solutions in goal models and use OptiMathSat as the backend reasoner [10]. The framework proposed by Nguyen et al. is a general-purpose goal modeling framework with multi-objective optimization capabilities. This paper's focus is risk analysis, so the meta-model includes concepts and relations from the risk modeling domain.

III. Risk Modeling

We follow the three layered approach of Asnar et al. [9] for modeling risk in goal models, and we keep the names for the sake of convention: asset layer, event layer, and treatment layer. These layers are shown in green, red, and blue, respectively, in our meta-mode presented in Fig. 1. Details of the meta-model are explained below.

A *situation* represents a partial state of the world [18]. In our goal modeling framework, situations are used as binary propositions that either hold *true* or *false*. *Goals* are desired situations. Stakeholder goals are represented in the asset layer. Goals without any parents (top goals) are *business objectives* that have associated *rewards* with them. The reward of a business objective is the utility gained from the achievement of the objective. Once the business objectives are identified, they are refined into more concrete child goals. *Refinement* nodes aggregate child goals and connect them with a parent goal. A parent goal may have multiple refinements, and it is satisfied when at least one of its refinement nodes is satisfied. A refinement node is satisfied when all child goals associated with the refinement goals are satisfied. Finally, the leaf goals with no incoming edges from refinement nodes are *tasks* that have associated *costs*. If a goal is labelled *mandatory*, it must be satisfied by any proposed solution. Otherwise, a goal is *preferred* and is to be satisfied by a solution if it is not in conflict with any elements of that solution.

To illustrate our modeling framework, we model the Loan Origination Process (LOP) [9] in Fig. 2. The process is initiated by a loan application and ends with a decision. The bank's ultimate motive is, of course, to earn income, and this is to be achieved by handling loan applications, and ensuring loan repayment. The topmost layer is the asset layer, goals are depicted as oval nodes, while refinement nodes are shown as black-filled circles. Child nodes are connected to refinement nodes, which then relate them to parent nodes. One example is G7: Ensure loan repayment, which is refined in to G8: Ask mortgage and G9:Monitor usage of loan. We omit numerical values from the model presented for visual simplicity. If the reward of all top goals are the same, the attribute values can be omitted. Similarly, all leaf goals (tasks) have a cost, which can be filled with relative (for example, within a scale of 1–5) or absolute values (230 for G9).

Risks, *anti-goals*, and other situations that lead to risks are modeled in the event layer. A situation models a partial state of the world [18]. An anti-goal is an undesired situation of the system being modeled, yet it is desired by another system, which may be malicious, such as S2: Forgery from external attack presented in an octagon node in Fig. 2. Similar to goal refinement, an *anti-goal refinement* links a set of child anti-goals to their parent anti-goal. A parent anti-goal may have multiple refinements as alternative ways of being achieved. The parent anti-goal is achieved when at least on of its refinements is achieved. A refinement is achieved when all child anti-goals connected to the refinement are achieved. For example, in Fig. 2 S2: Forgery from external attack is refined into S3: Information system is phished and S4: Credentials are obtained. A *risk* is a situation that harms one or more goals of the system being modeled with a possibility of loss. Each risk has an associated *likelihood* as the probability of the risk to happen. An *intentional risk* rises due to an anti-goal. In our illustrative example the risk S12: Electronic application is forged exists due to the anti-goal S2: Forgery from external attack. The directed edge with dotted line and black arrowhead from the anti-goal to the risk indicates an *increase likelihood* relation. Increase likelihood relation is only allowed between and anti-goal an a risk. *Incidental risks* are caused by external factors

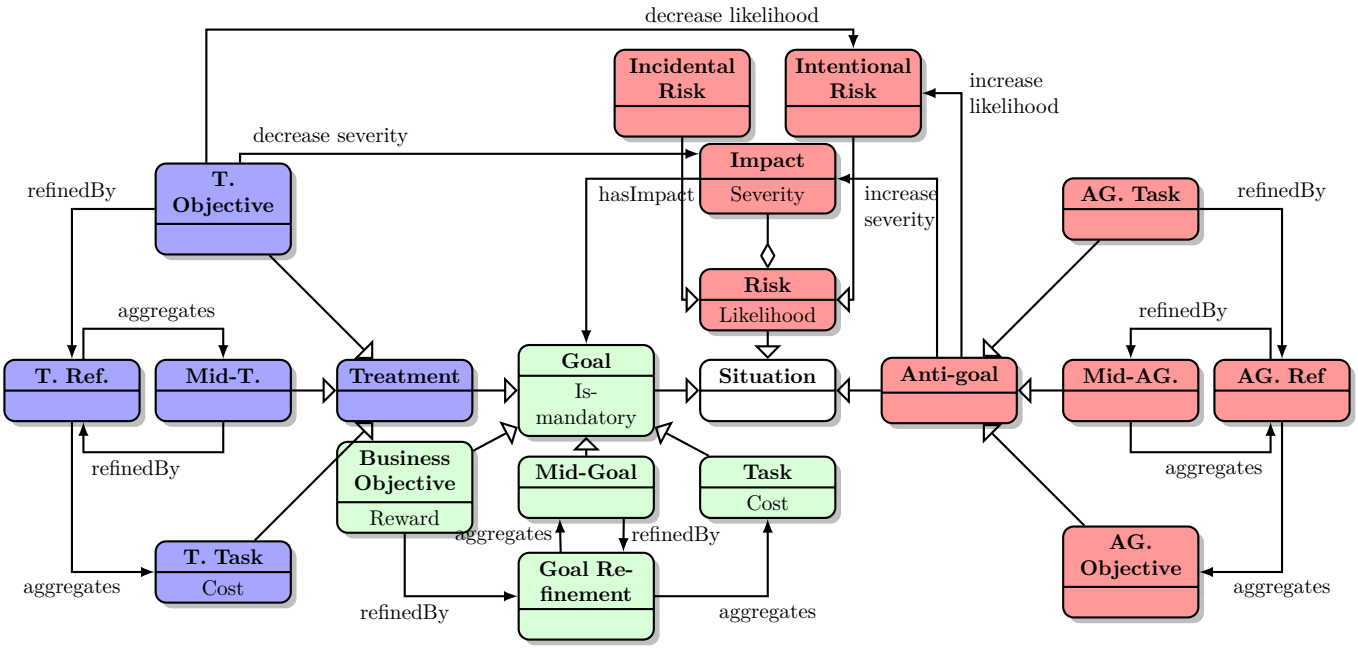


Figure 1: Meta–model for risk modeling in goal models

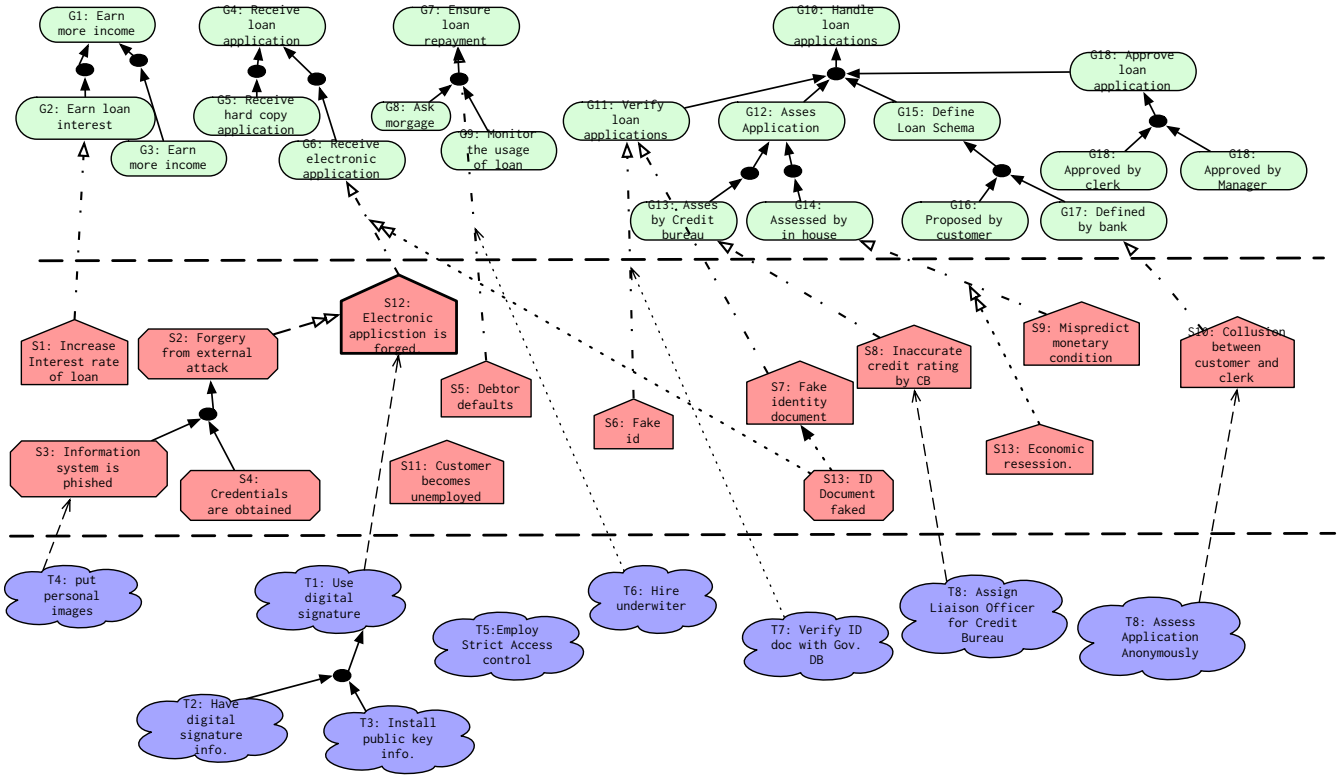


Figure 2: Illustrative example: Loan Origination Process (LOP). See Fig. 3 for the legend.

that can't be controlled, such as S13: Economic recession. *Impact* captures the effect of a risk on a goal. A risk may have different impacts on different goals. During modeling, the *severity* of the impact can be modeled in terms of absolute values such as monetary loss, or in terms of a relative scale can be used such as the five value scale used by CORAS [19]. In our illustrative example, S12: Electronic application is forged has an impact on the goal G6: Receive electronic application. In order to reduce the number of model elements, we omit the impact node between a risk and a goal, and use a directed dashed edge with a white arrowhead to represent *has-impact* relation. An anti-goal may increase the severity of an impact on a goal through *increase severity* relation. In LOP example S13: ID Document faked anti-goal increases the impact of S12 on G6.

Treatment layer includes *treatments* which are goals intended to mitigate the impact and/or reduce the probability of the risk. *Treatment refinement* links child treatments to parent treatment. Similar to other types of refinements, the parent treatment may have multiple refinements, each indicating an alternative way of achieving the parent. *Treatment objectives* are top treatments, and *treatment tasks* are the leaf treatments and they have a treatment cost attribute. Treatments mitigate the severity of an impact via *decrease severity* to recover after the occurrence of a risk. On the other hand tretments may also aim for prevention, decreasing the likelihood of the risk through *decrease likelihood* relation. In the example presented in Fig. 2 the treatment layer is the bottom layer in which treatment are presented in hexagonal nodes. T1:Use digital signature treatment decreases the likelihood of S12: Electronic application is forged (prevention). T6:ire underwriters reduces the severity of the impact of S5: Debtor defaults on G7: Ensure loan repayment.

Visual Notation. We follow the guidelines presented by Moody [20] for 'good' graphical notation. First, we use the horizontal position (planar variable) to distinguish layers, thick dashed lines clearly separates the three layers. The topmost layer is the asset layer, where stadium shaped nodes are goals, black filled dots are goal refinements and directed straight edges with black arrows are aggregation links of child goals to refinement nodes, and refines relations from the refinement nodes to parent goals. The straight edge is overloaded, yet the usage is inline with the conventional use, successfully conveys the semantics, yet decreases the visual clutter. Another symbol overload in this layer is the representation of goals where we do not differentiate the symbols for business objectives, middle goals, and tasks (top, middle, and leaf goals, respectively). Again, in this situation the symbol overload does not cause ambiguity for it is easy to differentiate the concepts by checking the incoming and outgoing edges. Among these three constructs two of them have numerical attributes. We propose placing the values in square brackets after the text when modeling on paper or a board, and ability to toggle the display of the values when modeling with a software tool to keep visual simplicity.

The middle layer is the event layer. We keep the convention of using house-shaped nodes for representing risk. A semantically immediate solution would be to use fire shaped nodes for risks, however this reduces the text area available for the description, and might introduce difficulty when using analog tools. Instead we opt for red color to signify danger. Even though incidental and intentional risks can be differentiated by perceptual configuration due to the incoming edges from anti-goals, we increase the thickness of the border of incidental risks (use as a retinal variable) to emphasize the distinction. Anti-goals are represented as parallelograms in [15], we prefer using octagons for the text are can be used more effectively. For anti-goal refinement we use the same visual elements as the goal refinements to preserve graph economy by keeping the number of graphical notations low. Dashed edges with white double triangle arrowheads are used to represent the increase likelihood relation (from anti-goals to risk nodes), double triangle arrowhead signifies an increase and increases the visual distance from other edge types together with the line type and arrowhead color. Increase severity relation is represented by dotted edges with white double triangle arrowheads. Although the visual distance is one (line type) between the representation of this relation and the former one, target constructs are different (the target of the former is a risk node, whereas the latter aims to has impact edge), helping perceptual configuration. Has impact relation is represented by dashed and dotted line with a white triangle arrowhead, for the white the arrowhead is a construct from the event layer, yet this relation does not signify an increase so a single triangle is chosen. According to the principle of dual coding, complementing the graphical notation with text annotations convey information more effectively, yet using edge labels creates usual clutter, we propose using tool-tips for edge labels for software tool support.

The bottom layer is the treatment layer. We use cloud shape nodes for the conventional octagonal nodes used in [9] is from the same shape family (polygons) as nodes used in the event layer therefore they are too similar. We use the color blue to fill the nodes to smooth out the danger implied by the nodes in the event layer. There are two inter-dependencies whose sources are treatments, one is decrease likelihood whose target is a risk node and the other is decrease severity whose target is an impact edge. We use open triangle arrow heads to emphasize the decrease, vary the line types to increase the visual distance. Fig. 3 summarizes our visual notation.

Three layered approach also helps managing the complexity of the models, enabling modularization, thus focusing on specific layers both in analog and digital tools. Considering all edge and node types, our notation exceeds the cognitive limit of six categories, yet this is an open problem present in other graphical notations [21].

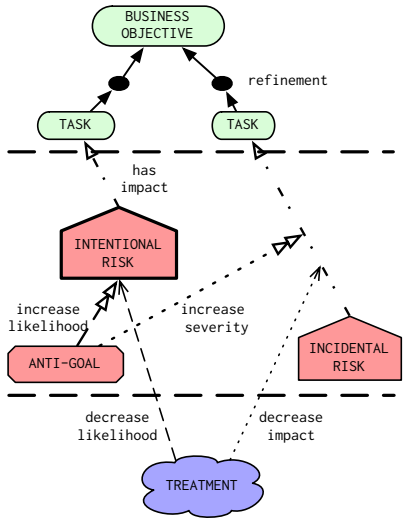


Figure 3: Legend for the visual notation

IV. Risk Analysis

Risk models are instrumental to comprehend, capture, and convey goals of a system, risks that may harm these goals, anti-goals that may create these risks or increase the severity of their impact on system goals, and treatments that prevent or mitigate the risks. The structure of the models enables systematic analysis to extract information that is not trivial for the analyst. Risk analysts are interested in multiple questions to asses and manage risks. Risk assessment includes setting likelihood and impact of each risk, which is done during modeling, and calculating the risk factor as the product of these two values. Once the risk assessment is done, the analyst has several strategies to manage risk. The first strategy is to avoid risk, that is, selecting a solution with the minimal risk factor. The second strategy is to prevent risk, that involves utilizing treatments to decrease the likelihood of the risk (or prevent the risk from happening). The third strategy is to mitigate the impact of the risk through treatments. Apart from the risk management strategies, the analyst may be interested in the reward of the system-to-be, giving a higher priority to utility rather than risks. The budget for the project is a constraint of which the analyst should keep track, so the analyst may opt for minimizing the cost, trading cost for risk factor. Below we provide a list of questions that help risk analyst to asses risk and select a strategy for risk management.

Q1. Which solution has the maximum goal rewards?
Q2. Which solution has the minimum task costs?
Q3. Which solution has the minimum risk factor?
Q4. Which solution has the minimum risk likelihood?
Q5. Which solution has the maximum recovery?
Q6. Which solution has the maximum prevention?
Q7. Which solution has the minimum treatment cost?

SMT/OMT reasoning is a powerful scalable approach that combines multi-objective optimization and satisfiability. We transform our models into SMT-LIB language [22] to pass the models as input to the external solver OptiMathSAT [17].

$$((\wedge_{j=1}^{n} G_j) \leftrightarrow R) \wedge (R \rightarrow G_{parent}) \tag{1a}$$

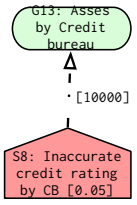
$$\text{Task Cost} = \sum_{G} ite(G_i, cost_{G_i}, 0) \tag{1b}$$

$$\text{Reward} = \sum_{G} ite(G_i, reward_{G_i}, 0) \tag{1c}$$

Formulas (1a) to (1c) presents the propositional encoding of the asset layer. Formula (1a) states that a refinement node R is achieved if and only if all subgoals that are connected to the refinement node is satisfied $(((\wedge_{j=1}^{n} G_j) \leftrightarrow R))$, and the satisfaction of the refinement R implies the satisfaction of the parent goal G. Formulas (1b) and (1c) are numeric pseudo-boolean functions that define the Task Cost and Reward functions, respectively. $ite(G_i, cost_{G_i}, 0)$ denotes an *if-then-else* term, which is evaluated as cost of the task G_i, if G_i is achieved, 0 otherwise.

The transformation to SMT-LIB is trivial for Formula (1a). In order to represent the if-then-else term for the cost and reward functions, soft–assertions are used with the syntax as the following, `(assert-soft (not G3) :weight 5 :id task.cost)`. This soft–assertion introduces a cost when `G3` holds `true`. For all task costs the same id (`:id cost`) should be used. In order to find the total task cost of a solution, we declare a task cost function, which is the sum of the cost attributes of all tasks that holds true within a solution. This is achieved *i.* declaring a real function `(declare-fun taskcost () Real)`. *ii.* asserting the id of the task costs as the value of this function `(assert (= reward (- task.cost 0)))` (Formula (1b)). In order to find the solution with the minimum task cost `(minimize taskcost)` command is used. Similarly, to find the solution with the maximum reward, we define a reward function that returns the reward of the business objectives that are assigned to be `true` in a given solution, achieved by the following two statements: `(declare-fun reward () Real)` `(assert (= reward (- bo.reward 0)))`. The optimization command is stated as `(maximize reward)`.

The previous paragraph explains how to formalize our models and provide the input to the reasoner to answer Q1 and Q2, that are general concerns when reasoning on goal models. The third question aims to help decision making in presence of risks. A risk factor is the product of the likelihood and the severity of the impact of a risk. A lower risk factor indicates a safer decision. Consider two tasks G13: Assessed by credit Bureau and G14: Assessed in house from the illustrative example which are connected to the same parent node through different refinement nodes, so they are alternatives to each other. G13 is under the risk of S8: Inaccurate credit rating by Credit Bureau with a likelihood



(a) Risk impacting a goal

$$risk.factor_{G_{13}} = 0.05 * 10000$$
$$rf_{G_{13},S_8} \leftrightarrow (G_{13} \wedge S_8)$$
$$\text{Risk Factor} = \sum_{G} ite(rf_{G_i}, risk.factorG_i, 0)$$

(b) Propositional Encoding

Figure 4: Risk impact on a goal

of 5% which may lead to a loss of €10000 whereas G14 is also under the risk of S9:Mispredict monetary condition with a likelihood of 10% and a possible loss of €8000. Out of these two alternative tasks G13 is safer due to its lower risk factor (€500 versus €800). Selecting a solution with the minimal risk factor is a result of adopting risk avoidance strategy for risk management.

Fig. 4a is a sample from the illustrated example presented in the previous section, Fig. 4b presents the propositional encoding of the Risk Factor objective function and how risk factor of an individual task is calculated. Once the risk factor is calculated for each impact on a task, the numerical value is presented in the SMT-LIB language similar to the presentation of task cost, however here we introduce an additional model element, which is set to true when both the task and the risk holds true in the model. The reason is that the analyst may choose the ignore some risks if she believes that they will not occur at run-time. So the calculated risk factor values are aggregated in the Risk Factor function only if the task is included in the solution where the risk is considered in the model by the analyst, as shown by the second line of Fig. 4b. In SMT-LIB, an individual risk factor is represented as `(assert-soft (not RF-G13-S8) :weight 500 :id task.risk.factor)`, and those individual values are aggregated by `(declare-fun riskfactor () Real) (assert (= riskfactor (- task.risk.factor 0)))`. Finally, the optimization scheme is set to `minimize riskfactor` to receive the solution that has the lowest risk factor.

Q4 searches for the solution that has the minimum chance of encountering a risk, regardless of the impact of the risk. Formula (3a) presents the associated risk likelihood of G_{13}. Similar to risk factor encoding, we introduce an additional element ($rl_{G_{13}}$) to ensure that only likelihood of risks that are considered by the analyst are included (Formula. (3b)). The soft-assertion `(assert-soft (not rl-G13) :weight 0.05 :id task.risk.likelihood)` is the translation of the if-then-else pseudo boolean function used in Formula (3c). The objective function returns the sum of the values for tasks that are included in the solution (tasks are set to be true). Here our assumption is that each impact edge comes from independent risks, therefore we sum the probabilities. More complex representation of risks in the event layer requires a change in the definition of this objective function in the future.

$$risk.likelihood_{G_{13}} = 0.05 \tag{3a}$$
$$rl_{G_{13},S_8} \leftrightarrow (G_{13} \wedge S_8) \tag{3b}$$
$$\text{Risk Likelihood} = \sum_{G} ite(G_i, risk.likelihood_{G_i}, 0) \tag{3c}$$

Q5 aims to find a solution including not only the stakeholder goals but also treatments to mitigate the impacts on the risks on goals. The mitigation of a treatment on an impact is represented by a decrease severity inter-dependency, and the numerical attribute indicates how much the treatment recovers the impact of the risk on a task. For the analysis, the important point is to achieve treatment objects that recovers tasks that are also included in the solution. From the illustrative example, it does not make sense to achieve the treatment 'verify ID documents with the government bodies' (T7) that decreases the severity of the impact to €5000 just because there is a risk of fake documents (S6) unless the goal of verifying loan applications (G11), which is affected by the risk, is part of the solution. Therefore we introduce a new element for each decrease severity relation, which is set to be true when the target treatment and the source risk are true, and the task that is under that risk is included in the solution (Formula (4a)). For all decrease severity inter-dependencies, the total recovery is aggregated by the pseduo-boolean function presented in Formula (4c). In order to avoid misleading solutions due to the numerical values, it is important to put a constraint during modeling a mitigation on an impact cannot be higher than the severity of the impact of the risk.

$$tr - risk - goal_{T_7,S_8,G_{13}} \leftrightarrow (G_{13} \wedge S_8 \wedge T_7) \tag{4a}$$
$$rec_{T_7,S_8} = 5000 \tag{4b}$$
$$\text{Recovery} = \sum_{T,G,S} ite(tr - risk - goal_{T_i,S_j,G_k}, rec_{T_i,S_j}, 0) \tag{4c}$$

Total or partial prevention of a risk (decreasing the likelihood) decreases *i.* the risk factor of affected goals. The purpose of asking Q6 is to discover solutions including goals and treatments that has the minimum total risk factor. The main difference between Q3 and Q6 is the former does not consider treatments so the focus is to find a solution within the asset layer with the lowest risk factor. On the other hand Q6 considers treatments which lead to different solutions within the asset layer, for example among two alternative solutions with risk factor 3 and 5 the solution for Q3 is the first alternative, however if there is a treatment that reduces the second risk factor to 2, the solution for Q6 is the second alternative. In order to answer Q6, we first need to calculate the likelihood of the

risk for each combination of treatments that decreases its likelihood. Then, for each new likelihood, a new risk factor is calculated for each task that is affected by that risk. Once pre-processing calculates these values, we transform the model, similar to the approached presented in Fig 4. The main difference is that, for each calculated value, a new element is created, which is set to true when all considered treatments, risk, and the task is true, and other treatments that prevents that risk but not considered are false. The pseudo-boolean function for the prevented risk factor aggregates the new risk factor values for the new elements that holds true.

Algorithm 1 creates necessary SMT/OMT formulas. Before running the reasoner, we need to declare all possible likelihoods of a risk resulting from combining different treatments that reduce its likelihood. Lines 3 to 5 in in Algorithm 1 calculates the likelihood after prevention by all possible combinations of treatments that are related to a given risk.

```
1  initialization;
2  foreach task t of asset layer do
3      foreach impact i of risk r on t do
4          M = {Treatments decreases likelihood of r}
           foreach N ⊂ M do
5              calculate new likelihood of r ;
6              calculate risk factor after prevention;
7              declare a new element n;
8              n ↔ (t ∧ r ∧ {⋀ treatments ∈ N} ∧
                 ¬{⋁ treatments ∈ M \ N});
9              assert risk factor after prevention for n;
10         end
11     end
12 end
```

Algorithm 1: Minimizing total risk factor with prevention

Q7 focuses on the objective of minimizing the cost of treatments selected as part of the solution. The transformation from the models to SMT/OMT clauses are quite similar to the transformation of the asset layer. Formula (5a) presents the cost function for treatments.

$$\text{Treatment Cost} = \sum_{T} ite(T_i, cost_{T_i}, 0) \tag{5a}$$

Custom objective functions. Risk analysis plays an important role in project management for projects of all sizes from IT projects to governmental decisions. There are two main strategies to follow for risk management, the first strategy is to avoid risk, selecting solutions that have low risk factor. An analyst who is interested in risk avoidance tries to answer Q3. The second strategy is to accept risk, but either prevent it from happening (Q6) or mitigate its impact (Q5). Cost is an important factor for deciding the risk management strategy. In some cases risk prevention is cheaper than recovery, for example preventive health measures are usually cheaper, that's the reason behind governments' investments in public vaccination campaigns to prevent an epidemic so that they would not need to face with the cost of treating infected masses. In some other cases, the opposite is true. Instead of keeping several backup servers all the time to prevent any service disruption, a bank may prefer to pay for a recovery service in case of a down-time. Safety critical systems give the highest priority to prevent risks, so they ask Q4 and Q6 first, and may give cost a lower priority. The analyst may also give the utility gained from the achievement of goals a high priority. A start-up may choose to roll a product to the market even though such move is associated with a high risk factor, yet the reward of their business objectives is too tempting. A thorough risk analysis requires combining these analyses, such as finding the solution with the minimal total cost of treatments and goals that has the minimal risk factor with maximum risk recovery. Depending on the objectives of the analysts several queries may be constructed.

Using SMT/OMT reasoning the analyst may set the optimization scheme for the solution in three ways. When there is a clear ordering of objectives, the analyst may opt for lexicographic optimization. The reasoner orders the solution according to the order of objectives, so the optimum solutions with respect to the first objective is returned first. If there are multiple solutions with the same objective value, the solutions are then ordered with respect to the second objective and so on. An example order of objectives is the following.

```
(minimize taskcost)
(minimize task.risk.factor.prevented)
(maximize reward)
```

The analyst may also construct a single multi-objective optimization function such as $0.6 * \text{task.cost} + 0.4 * \text{treatment.cost}$ and minimize or maximize that function. The third option is to combine these techniques together, having multiple objective functions ordered clearly.

Additional constraints. Constrained goal models and SMT/OMT reasoning allow applying additional constraints on solutions. For example, it is possible to set a constraint on the total budget of the solution, where the total budget is the sum of task and treatment costs. The reasoner returns the solution with maximal recovery whose total cost is less than €100000 in the following example. Another example is for the consideration of risks, the analyst may choose to ignore risks whose likelihood is under a certain threshold, or allow risk factor up to a certain value. Additional constraints can be asserted as both hard and soft constraints in SMT-LIB.

```
(assert < totalCost 100000)
(maximize recovery)
```

Security risk analysis. Anti-goals [15] and obstacle analysis [23] are used to analyze risk that is caused by

malicious actors. existing reasoning techniques can be applied in the event layer to determine whether a risk holds or not. In this paper our focus is on finding optimal solutions with respect to loss, cost, and rewards.

Opportunity analysis. An opportunity is a probabilistic situation that has a positive impact on stakeholder goals. Opportunity identification and analysis is conducted during new product development [24], our approach can be used to discover optimal solutions with respect to gain. Opportunity factor is calculated similar to risk factor, and it is desired to be maximized.

Tool support. We have implemented a proof-of-concept tool to demonstrate our approach. The tool is a standalone application based on Eclipse Modeling Framework. The graphical models are translated into SMT-LIB language, which are then provided to the SMT/OMT solver. The results retrieved by the solver are highlighted in the graphical model, and a report on the solution is provided.

Scalability. We set up an experiment to test the scalability of our approach, running the experiment run on a Windows 64 bit machine with Intel(R) Core(TM) i7-3770 CPU 3.40Ghz and 8GB of RAM, and collected the reasoning time reported by OptiMathSAT to find the solution. For the experiments, we use OptiMathSAT version 1.3.10. Starting from an initial input model of 22 elements, we kept replicating the input model and connecting to the same parent node. The solver returned the results for the largest model with 17275 model elements uder 400 miliseconds showing a linear trend. Our results are parallel to those reported in [10], [17].

V. Discussion and the Related Work

Asnar et al. [9] propose Goal–Risk (GR) Framework for modeling and reasoning about risk during requirements engineering process. The models have three layers for goals, events, and treatments. The framework provides extended versions of label propagation algorithms presented in [25] and finds optimal solutions with respect to a single objective such as cost. Our framework provides multi-objective optimization, so the systematic reasoning can answer more questions related to solutions with respect to risk factor, risk prevention, and risk mitigation.

KAOS [12], *i** [13], NFR [11], and Tropos [14] are goal modeling frameworks for general requirements engineering purposes. KAOS is later extended with anti-goals [15] to explicitly model malicious stakeholder goals and obstacles [26] to model incidental risks. Mayer et al. [27] model business and IT assets and the corresponding security goals and requirements as an extension of *i** [13]. Countermeasures are applied to mitigate risks to reduce any impact of the risks on business assets. Liu et al. [7] propose a framework to analyze security and privacy requirements based on *i** [13] and NFR framework [11]. The focus of their analysis is to explore alternative solutions based on threats, vulnerabilities, and countermeasures. Secure Tropos [28] is a formal framework based on Tropos to model security requirements. Matulevicius et al. extend Secure Tropos syntax and semantics to handle security risks in [29]. Recently Siena et al. [30] model risks in open source software in terms of situations, risks, and goals and use label propagation algorithms to check whether the goals are satisfied. Their models do not include treatments for risks. Later Costal et al. [31] combine *i** and RiskML [30] to align business goals and risk in open source software and use existing reasoning methods [32] to analyze the achievement of stakeholder goals. The reasoning focuses on propagating the impact of risks to related goals. None of these approaches find optimal solutions, their focus is whether satisfaction of goals is affected by risks. Our approach is complimentary to these, our focus is to discover high-risk solutions, solutions within a limited budget, solutions that maximize prevention and mitigation. Pitangueria et al. [33] use OMT–based reasoning to find a risk-aware solution for the multi-objective next release problem [34], yet they do not use goal models or any hierarchically structured representation of requirements, so they do not capture and reason on the hierarchy and the interdependencies among requirements.

Feather et al. [35] propose Defect Detection and Prevention (DDP) as a three layered (objectives, risks, and mitigation) approach similar to GR Framework. Each risk has a likelihood as the probability of occurrence, and the severity of a risk is represented by an impact relationship between the risk and an objective. Our meta-model is more expressive than DDP models, capturing the interdependencies among treatments and risks.

CORAS [19] is a risk analysis framework that models, analyzes, and treats risks. In CORAS, each risk is analyzed independent of system or malicious actor objectives. Each risk has a single impact and likelihood, so the effect of the same event on multiple objectives are considered as separate risks. CORAS also does not find optimal solutions with respect to multiple objectives.

The meta-model includes multiple attributes for costs, rewards, likelihoods, and other values which are important to find the optimal solutions within a model with respect to the optimization scheme. Finding the absolute or relative values for these attributes are beyond the scope of this paper, Saaty [36] provides a background on how to set these values in hierarchic structures, Liaskos et al. [37] demonstrates this technique on goal models. Delphi method [38] is another method that can be used by a group of analysts to iteratively set those values and reach an agreement.

VI. Conclusions

This paper presents a multi-objective goal-oriented risk modeling and analysis framework by extending Asnar et al. [9] with constrained goal models [10]. We provide a meta-model enhanced with inter-dependencies in the model elements that determines the optimality of a

solution. Our visual notation follows the design principles summarized in [20]. We transform goal–oriented risk models to SMT/OMT to discover optimal solution with respect to multiple objectives. Our proof-of-concept tool automatically maps the models into SMT/OMT formulas and retrieves solutions from the back–end reasoner. We investigate a pre-defined set of questions for analyzing risk yet our approach is flexible so that the analyst may define her own objective function and search for the optimal solution with respect to this custom objective function.

Limitations. SMT/OMT–based reasoning can handle linear arithmetic over rationals, it is a limitation for our framework. We overcome this limitation by pre-processing our models before transformation to SMT-LIB formulas, yet using combinations of treatments increases the number of formulas. Our graphical models suffer from the complexity problem of goal models. Building large goal models is an error-prone manual activity where the modeller loses grasp of what her model says as it gets bigger. We use layers to decrease the number of visible elements when the focus is on a single layer.

Future Work. We intend to further develop our proof-of-concept modeling tool into a prototype to in order to conduct usability analysis for the tool and validation for our overall approach. We plan to validate our approach with a real case study in the near future. The conceptual model provided can be extended with actors as well as social interactions, to support reasoning on global as well as local models.

Acknowledgements. This research was partially supported by the ERC advanced grant 267856 'Lucretius: Foundations for Software Evolution'. This research has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 653642 - VisiON.

References

[1] B. Flyvbjerg and A. Budzier, "Why your it project may be riskier than you think", *Harvard Business Review*, vol. 89, no. 9, pp. 601–603, 2011.

[2] A. P. Sage and Y. Y. Haimes, *Risk modeling, assessment, and management*. John Wiley & Sons, 2015.

[3] G. Roy and T. Woodings, "A framework for risk analysis in software engineering", in *Proceedings Seventh Asia-Pacific Software Engeering Conference. APSEC 2000*, Dec. 2000, pp. 441–445.

[4] S. Cornford, M. Feather, V. Heron, and J. Jenkins, "Fusing quantitative requirements analysis with model-based systems engineering", in *14th IEEE International Requirements Engineering Conference (RE'06)*, Sep. 2006, nil.

[5] J. Horkoff and E. Yu, "Analyzing goal models", in *SAC*, 2011.

[6] E. Paja, F. Dalpiaz, and P. Giorgini, "Modelling and reasoning about security requirements in socio-technical systems", *Data & Knowledge Engineering*, vol. 98, no. nil, pp. 123–143, 2015.

[7] L. Liu, E. Yu, and J. Mylopoulos, "Security and privacy requirements analysis within a social setting", in *Journal of Lightwave Technology*, Sep. 2003, pp. 151–161.

[8] P. Giorgini, F. Massacci, J. Mylopoulos, and N. Zannone, "Requirements engineering for trust management: Model, methodology, and reasoning", *International Journal of Information Security*, vol. 5, no. 4, pp. 257–274, 2006.

[9] Y. Asnar, P. Giorgini, and J. Mylopoulos, "Goal-driven risk assessment in requirements engineering", *Requirements Engineering*, vol. 16, no. 2, pp. 101–116, 2010.

[10] M. C. Nguyen, R. Sebastiani, P. Giorgini, and J. Mylopoulos, "Multi-objective reasoning with constrained goal models", http://arxiv.org/abs/1601.07409, 2016.

[11] J. Mylopoulos, L. Chung, and B. Nixon, "Representing and using nonfunctional requirements: A process-oriented approach", *IEEE Transactions on Software Engineering*, vol. 18, no. 6, pp. 483–497, 1992.

[12] A. Dardenne, A. van Lamsweerde, and S. Fickas, "Goal-directed requirements acquisition", *Science of Computer Programming*, vol. 20, no. 1-2, pp. 3–50, 1993.

[13] E. Yu, "Towards modelling and reasoning support for early-phase requirements engineering", in *3rd IEEE International Symposium on Requirements Engineering*, Jan. 1997, pp. 236–235.

[14] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos, "Tropos: An agent-oriented software development methodology", *Autonomous Agents and Multi-Agent Systems*, vol. 8, no. 3, pp. 203–236, 2004.

[15] A. Van Lamsweerde, "Elaborating security requirements by construction of intentional anti-models", in *Proceedings of the 26th International Conference on Software Engineering*, IEEE Computer Society, 2004, pp. 148–157.

[16] R. Sebastiani and S. Tomasi, "Optimization modulo theories with linear rational costs", *ACM Transactions on Computational Logic*, vol. 16, no. 2, pp. 1–43, 2015.

[17] R. Sebastiani and P. Trentin, "Optimathsat: A tool for optimization modulo theories", in *Computer Aided Verification*, Springer Science + Business Media, 2015, pp. 447–454.

[18] T. Wetzel, *States of affairs*, http://plato.stanford.edu/archives/fall2008/entries/states-of-affairs/, Last accessed Fri Jan 29 13:40 2016, 2003.

[19] R. Fredriksen, M. Kristiansen, B. A. Gran, K. Stølen, T. A. Opperud, and T. Dimitrakos, "The coras framework for a model-based risk management process", in *Computer Safety, Reliability and Security*, ser. Computer Safety, Reliability and Security. Springer Science + Business Media, 2002, pp. 94–105.

[20] D. Moody, "The physics of notations: Toward a scientific basis for constructing visual notations in software engineering", *IEEE Transactions on Software Engineering*, vol. 35, no. 6, pp. 756–779, 2009.

[21] D. L. Moody, P. Heymans, and R. Matulevicius, "Improving the effectiveness of visual representations in requirements engineering: An evaluation of i* visual syntax", in *2009 17th IEEE International Requirements Engineering Conference*, Aug. 2009, pp. 171–180.

[22] C. Barrett, P. Fontaine, and C. Tinelli, "The smt-lib standard: version 2.5", Department of Computer Science, The University of Iowa, Tech. Rep., 2015, Available at `www.SMT-LIB.org`.

[23] L. Duboc, E. Letier, and D. S. Rosenblum, "Systematic elaboration of scalability requirements through goal-obstacle analysis", *IEEE Transactions on Software Engineering*, vol. 39, no. 1, pp. 119–140, 2013.

[24] P. A. Koen, "The fuzzy front end for incremental, platform and breakthrough products and services", *PDMA Handbook*, pp. 81–91, 2004.

[25] P. Giorgini, J. Mylopoulos, and R. Sebastiani, "Goal-oriented requirements analysis and reasoning in the tropos methodology", *Engineering Applications of Artificial Intelligence*, vol. 18, no. 2, pp. 159–171, 2005.

[26] A. van Lamsweerde and E. Letier, "Handling obstacles in goal-oriented requirements engineering", *IEEE Transactions on Software Engineering*, vol. 26, no. 10, pp. 978–1005, 2000.

[27] N. Mayer, E. Dubois, and A. Rifaut, "Requirements engineering for improving business/it alignment in security risk management methods", in *Enterprise Interoperability II*, ser. Enterprise Interoperability II. Springer Science + Business Media, 2007, pp. 15–26.

[28] P. Giorgini, F. Massacci, J. Mylopoulos, and N. Zannone, "Modeling security requirements through ownership, permission and delegation", in *13th IEEE International Conference on Requirements Engineering (RE'05)*, - 2005, nil.

[29] R. Matulevicius, H. Mouratidis, N. Mayer, E. Dubois, and P. Heymans, "Syntactic and semantic extensions to secure tropos to support security risk management", *J. UCS*, vol. 18, no. 6, pp. 816–844, 2012.

[30] A. Siena, M. Morandini, and A. Susi, "Modelling risks in open source software component selection", in *Conceptual Modeling*, ser. Conceptual Modeling. Springer Science + Business Media, 2014, pp. 335–348.

[31] D. Costal, L. López, M. Morandini, A. Siena, M. C. Annosi, D. Gross, L. Méndez, X. Franch, and A. Susi, "Aligning business goals and risks in oss adoption", in *Conceptual Modeling*, ser. Conceptual Modeling. Springer Science + Business Media, 2015, pp. 35–49.

[32] P. Giorgini, J. Mylopoulos, E. Nicchiarelli, and R. Sebastiani, "Reasoning with goal models", in *Conceptual Modeling - ER 2002*, ser. Conceptual Modeling - ER 2002. Springer Science + Business Media, 2002, pp. 167–181.

[33] A. Pitangueira, P. Tonella, A. Susi, R. Maciel, and M. Barros, "Risk-aware multi-stakeholder next release planning using multi-objective optimization", in *REFSQ*, 2016.

[34] Y. Zhang, M. Harman, and S. A. Mansouri, "The multi-objective next release problem", in *GECCO*, 2007, pp. 1129–1137.

[35] M. Feather, "Towards a unified approach to the representation of, and reasoning with, probabilistic risk information about software and its system interface", in *15th International Symposium on Software Reliability Engineering*, Nov. 2004, pp. 391–402.

[36] T. L. Saaty, "How to make a decision: The analytic hierarchy process", *European Journal of Operational Research*, vol. 48, no. 1, pp. 9–26, 1990.

[37] S. Liaskos, R. Jalman, and J. Aranda, "On eliciting contribution measures in goal models", in *2012 20th IEEE International Requirements Engineering Conference (RE)*, IEEE, Sep. 2012, pp. 221–230.

[38] H. A. Linstone, M. Turoff, *et al.*, *The Delphi method: Techniques and applications*. Addison-Wesley Reading, MA, 1975, vol. 29.