An Approach to Specify and Analyze Goal Model Families

Azalia Shamsaei[1], Daniel Amyot[1], Alireza Pourshahid[1], Edna Braun[1], Eric Yu[2], Gunter Mussbacher[3], Rasha Tawhid[4], and Nick Cartwright[5]

[1] School of Electrical Eng. and Computer Science, University of Ottawa, Canada
{asham092,apour024}@uottawa.ca,
{damyot,ebraun}@eecs.uottawa.ca
[2] Department of Computer Science, University of Toronto, Canada
eric@cs.toronto.edu
[3] Department of Systems and Computer Engineering, Carleton University, Canada
gunter@sce.carleton.ca
[4] See the *in Memoriam* Section
[5] Independent Consultant, Canada
ncart@sympatico.ca

Abstract. Goal-oriented languages have been used for years to model and reason about functional, non-functional, and legal requirements. It is however difficult to develop and maintain these models, especially when many models overlap with each other. This becomes an even bigger challenge when a single, generic model is used to capture a family of related goal models but different evaluations are required for each individual family member. In this work, we use ITU-T's Goal-oriented Requirement Language (GRL) and the jUCMNav tool to illustrate the problem and to formulate a solution that exploits the flexibility of standard GRL. In addition, we report on our recent experience on the modeling of aerodrome regulations. We demonstrate the usefulness of specifying families of goal models to address challenges associated with the maintenance of models used in the regulatory domain. We finally define and illustrate a new tool-supported algorithm used to evaluate individual goal models that are members of the larger family model.

Keywords: Goal Modeling, Goal-oriented Requirement Language, Key Performance Indicator, Legal Compliance, Tools, URN, Variability.

1 Introduction

Goal-oriented modeling has been used successfully in the past to measure compliance of business processes with regulations, to measure business process performance, as well as to analyze organizational security requirements [1,2]. However, can it handle the modeling of complex regulations that apply to multiple types of organizations?

There are many regulations to be modeled, and these models can be used for compliance assessment by regulators. However, different regulatory requirements

Ø. Haugen, R. Reed, and R. Gotzhein (Eds.): SAM 2012, LNCS 7744, pp. 34–52, 2013.

apply to different types of organizations. To have one goal model per type of organization would require significant maintenance effort when the legal context evolves, with the additional risk of introducing inconsistencies and other types of errors in the model. In this context, we should explore ways to capture a family of goal models, whose individual members can be extracted for compliance analysis. Our hypothesis is that a goal language can be tailored to support the concept of a *model family*, and hence mitigate the risks of inconsistencies while minimizing maintenance effort.

There are many existing goal languages and most have been used in one way or another in a legal compliance context [1]. However, in this paper, we selected the Goal-oriented Requirement Language (GRL) for several reasons:

1. GRL is standardized as part of the User Requirements Notation (URN) [3,4];
2. the extensions to GRL that support the concept of Key Performance Indicator (KPI) [5] are useful to measure compliance in units that people doing inspection/audit activities can actually understand;
3. it is possible to tailor (i.e., *profile*) the language through metadata, URN links and constraints;
4. a tool that supports these concepts is available [6] and enables the creation of evaluation algorithms [7] that exploit these concepts.

In this paper, an example related to aerodrome regulations is described in section 2, followed by requirements for the support of families of goal models. In section 3, we report on a first attempt to model regulation families with GRL and KPIs, and show limitations. In order to solve these limitations, we introduce in section 4 (our core contribution) a GRL profile including OCL well-formedness constraints as well as a new propagation algorithm, which are illustrated on the aerodrome regulations example. Related work, limitations, and future work items are discussed in section 5, while section 6 presents our conclusions.

2 Modeling Issues: Illustrative Example

With this example, we will illustrate why modeling and analyzing of regulations that apply to multiple types of organizations is challenging, and why a family of goal models could be used to address this issue. This example is inspired from realistic aerodrome security regulations, with a focus on perimeter signage and access control.

We model regulations starting with their high-level goals (e.g., perimeter security). We decompose these goals into operational and control rules. Furthermore, we define KPIs for each rule that measures their compliance level. Although this works well for compliance measurement [2], when different versions of the same model are used for various types of organizations, one runs into scalability and maintainability issues.

Perimeter Security: One of the important goals of an aerodrome is to provide effective perimeter security. The regulator issues regulations that establish

obligations on aerodrome operators and specify various security elements (e.g., signage, requirements for fencing, access control, etc.) that would comprise an appropriate system of perimeter security [8]. Due to the sensitivity of security regulations, we use a simplified and obfuscated example (Fig. 1) instead of a real one. Its structure however is illustrative and representative of the kind of issues faced while modeling real regulations.

GRL Modeling: We use GRL to model the regulations. GRL models (Fig. 1) consist of *intentional elements*, such as *goals* (, e.g., Perimeter Security). Intentional elements can be connected to each other using *contribution links* (→) with a quantitative weight ([-100..+100]). AND/OR *decomposition links* ( ) can be used to connect elements with sub-elements. *Dependency links* ( ) are used in our example to show that an intentional element depends on another one to satisfy a goal. We also use *resources* ( ) to capture conditions that goals depend on (e.g., in Fig. 1, Access control system rule1 depends on Condition). Finally, a *GRL strategy* describes a particular configuration of alternatives and initial satisfaction values in the GRL model, while a *GRL evaluation mechanism* propagates these values to the other intentional elements of the model (through their links) and compute their satisfaction values (Fig. 2: the values shown just above intentional elements). Different evaluation algorithms exist for GRL [7].

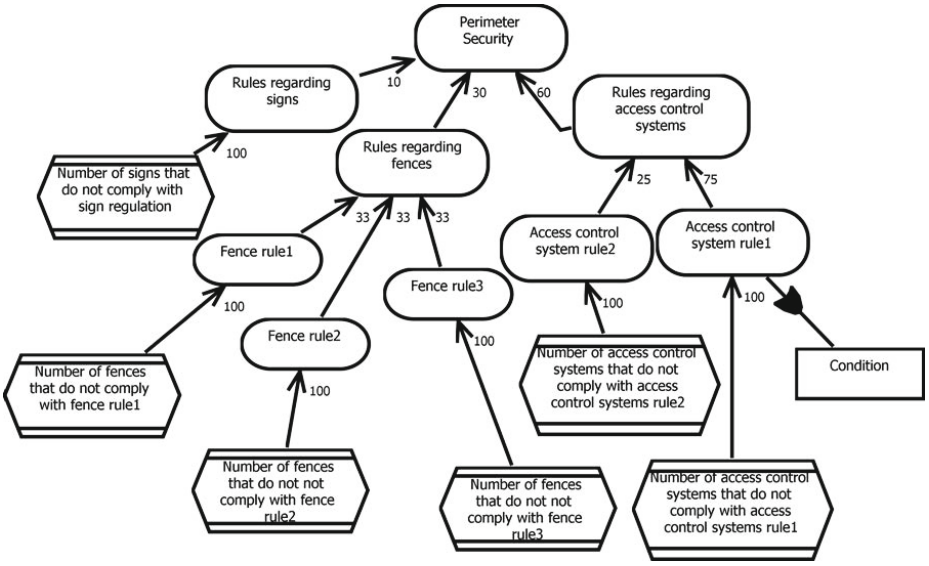

Fig. 1. Goal model of a perimeter security regulation (artificial example)

Key Performance Indicators (KPIs): *Indicators/KPIs* (, e.g., Number of fences that do not comply with fence rule2) in GRL are goal model elements that convert values from the real world (e.g., $45,000) to a GRL satisfaction level according to a defined conversion method (e.g., target, threshold, and worst-case

values). Using GRL strategies, one can initialize the KPI value sets manually or through external data sources (e.g., a Business Intelligence system). KPIs can be used to evaluate the satisfaction of regulation rules and ultimately the overall satisfaction level of the regulation or compliance level of the organization. KPIs can be linked with intentional elements only in the following, limited way. KPIs can be the source of a contribution link or decomposition link and can be used in dependency links. KPIs are included in the second version of the URN standard. GRL models with KPIs can be created, managed, and analyzed with jUCMNav [6], a free, Eclipse-based, open source tool that supports the URN standard and more, including the evaluation of the models with color-coded feedback (Fig. 2: the greener, the better, and the redder, the worse).[1]

Requirement for Families of Goal Models: Aerodromes are divided into various categories (or *types*), based on a set of factors. Some elements of the regulation are only applicable to specific types (e.g., to TYPE1, TYPE2, or TYPE3).

A regulator requires a generic model for each regulation rather than having separate models based on the aerodrome type. The generic model gives a holistic view of the regulation as opposed to a specific model for each type of aerodrome. For instance, out of 15 model elements related to fences, only 3 are specific to TYPE1 aerodromes. Therefore, if a separate model is created for TYPE1 aerodromes, the model will not show the complete picture of the regulation. Furthermore, having separate models for each type increases maintenance costs and the risk of errors in the models as different versions evolve. Hence, a technique is required to create a *family* of goal models, here based on aerodrome types, allowing some of the rules to be ignored or hidden in the generic model when a specific type of aerodrome is being evaluated.

3 Attempt 1: Modeling Families with Standard GRL

To meet the requirements for goal model families, we first tried to use the existing GRL capabilities, including dependency links. In GRL, the satisfaction value of the depender cannot be higher that the satisfaction value of the dependee. We attempted to model the aerodrome categories with GRL *resource* elements and *dependency links* from the rules. Some regulations can apply to more than one type. In Fig. 2, Access control system rule1 is restricted to aerodromes of types TYPE1 or TYPE2. We combine these two by OR-decomposing Condition1.

In addition, some rules are applicable only under certain conditions (independent of the type, and more dynamic by nature). If the conditions are true, then the rule should be imposed. As illustrated in Fig. 2, such a condition also applies to Access control system rule1 for aerodromes of type TYPE1. Yet again, an intermediate node (Condition2) is needed to handle the Condition, this time with an AND-decomposition.

After modeling the regulation, we define GRL strategies, each representing a global situation that indicates the rules that apply to each of the categories

[1] Please see the online version of this paper for color diagrams.

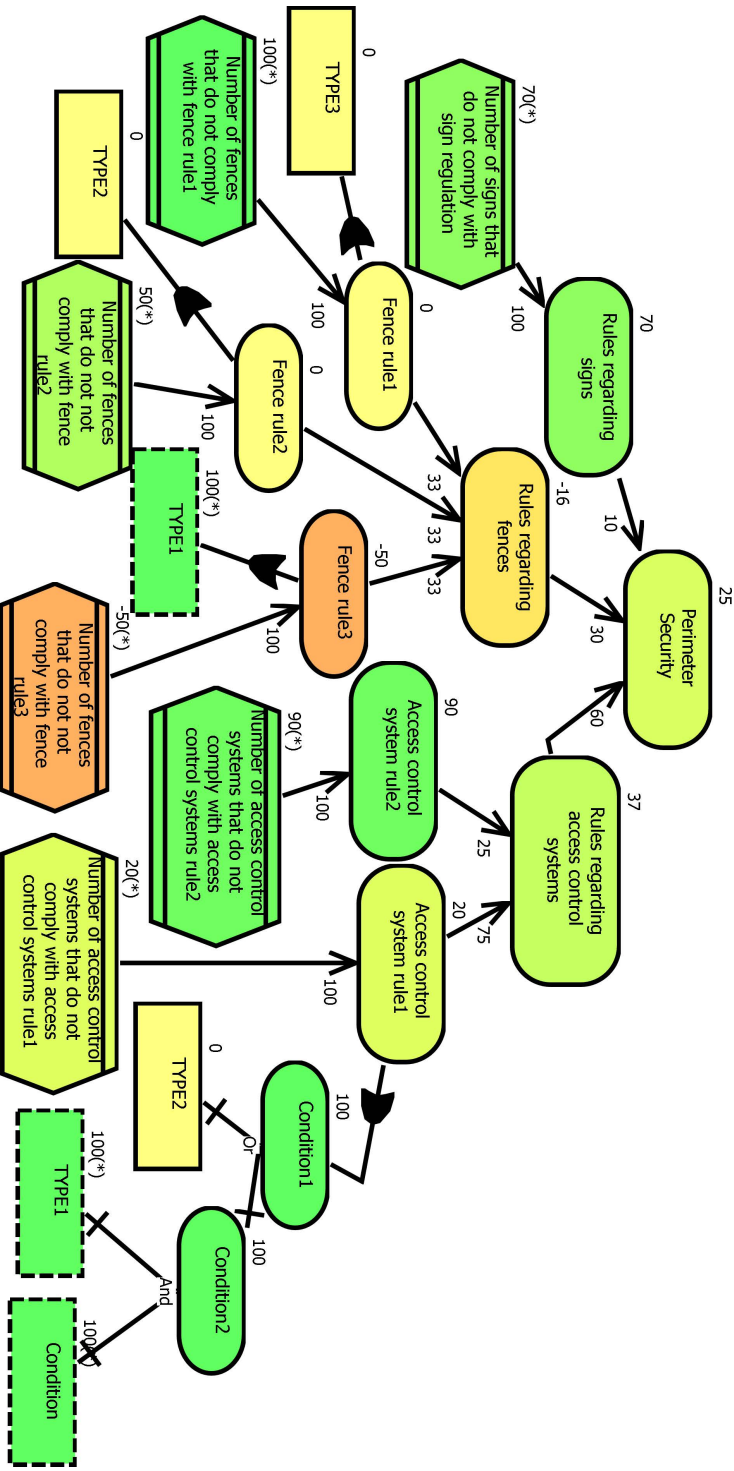

Fig. 2. Generic model evaluated against a TYPE1 strategy (artificial example)

(TYPE1, TYPE2, or TYPE3). Depending on the strategy, either 0 (not selected) or 100 (selected) is used as an initial satisfaction value for the type categories. Fig. 2 illustrates a strategy where the value of the TYPE1 resource element is 100 and the other resource elements (TYPE2 and TYPE3) values are 0. Therefore, all the elements dependent on TYPE2 or TYPE3 will be evaluated to a maximum value of 0 and will not affect the evaluation of the model. Hence, Fence rule1 and Fence rule2 are evaluated to 0, whatever the values of their KPIs, since they are not applied to TYPE1 aerodromes. In addition, Access control system rule1 is used in TYPE1 or TYPE2 aerodromes. Since resource TYPE1 has a satisfaction level of 100, and since the strategy explored here has the value of Condition set to 100, the satisfaction of Condition2 becomes 100 and therefore, Condition1 is also evaluated to 100, which means that Access control system rule1 will not be ignored. The rules that are not dependent on any type represent *all* categories and by default are not ignored in any of the strategies.

Using this method, we managed to create a family of goal models from the generic goal model. Although this approach addresses our requirement to some extent for simple examples, it proved not to be a good solution for more complex real-life examples. We identified four main problems:

1. Noise in the models;
2. Scalability and maintenance;
3. Ambiguity between the two types of conditions;
4. Evaluation and analysis.

First, an additional model element with a dependency link is connected to each rule to specify the category of the rules. In cases where a rule is valid for more than one type, more model elements and links are required. For instance, in the simple case where one rule is applicable to two types of aerodromes, we need three additional intentional elements to show this condition (Fig. 2: Condition 1 goal, TYPE1 condition, and TYPE2 condition). When the number of rules increases, there is much additional *noise* that is not really core to the model but only used to define conditions and categories. This problem can be somewhat mitigated by moving the conditions and categories to separate diagrams of a same model. Hence, although the model still includes the additional elements and links, the diagrams used by the end user for analysis purposes are much cleaner and manageable. However, since some rules may apply to more than one aerodrome type, the rules will be repeated in multiple diagrams, increasing their size as well as maintenance complexity.

Furthermore, there are often conditions other than categories (as illustrated in Fig. 2: Condition) that need to be considered more dynamically when the applicability of the rules is being assessed. If we use resources and dependencies for showing yet another type of conditions, the models with two types of condition nodes become ambiguous and too complex to be used by targeted users of this method (i.e., compliance officers).

Finally, the evaluation of these models using the existing GRL evaluation algorithms is often misleading. For example, if the contribution levels of the eliminated rules and applied rules are kept the same as in the generic model,

even with a satisfaction value of 100 for an applicable rule, the target intentional element will have a satisfaction value lower than 100, which in this application is interpreted as non-compliance. For example, if the satisfaction value of Fence rule1 is 100 and TYPE3 organizations are evaluated based on the GRL model in Fig. 2 (i.e., the satisfaction values of Fence rule2 and Fence rule3 are 0 or these model elements are even removed from the model), then the target intentional element Rules regarding fences will evaluate to 33 (and hence non-compliant). However, the satisfaction value of Rules regarding fences should be 100 (and hence compliant), because all rules for TYPE3 organizations are fully satisfied. Existing evaluation algorithms consider all the nodes and contribution links connected to a target intentional element while calculating its satisfaction level. Hence, there is an implicit requirement to redistribute the weight of removed contributions onto the weight of the contributions that remain.

4 Attempt 2: GRL with a Goal Model Family Profile

The URN standard, which includes GRL, offers lightweight mechanisms to extend the language. A GRL profile takes advantage of an important URN concept, namely *metadata*, which are name-value pairs used to annotate any model element (e.g., for stereotyping). Note that the metadata facilities in jUCMNav are generic, i.e., the tool allows an arbitrary set of stereotypes to be used for model annotation. Analysis algorithms can also take advantage of profile information. *Constraints*, expressed in OCL, can be used to enforce well-formedness constraints for GRL models with metadata and to query the GRL model. In this section, we introduce stereotypes (metadata) and a propagation algorithm for GRL model families, with a particular focus on compliance. The profile is called *measured compliance* profile.

4.1 Goal Model Family Concepts

In order to solve the noise, maintenance, and ambiguity issues of the models discussed in section 3, a new profile was created to manage goal model families within GRL. Fig. 3 introduces its basic concepts. In essence, a *family* (a GRL model) is composed of *model elements* (we focus on GRL intentional elements here, but obviously actors and links are included as well). Some of these elements can be *tagged* with metadata describing categories. These tagged model elements are part of *family members*, where a member is a subset of the family regrouping the elements for a given category. A model element can be tagged with multiple categories and hence can be part of many family members.

In our aerodrome context, we tag intentional elements with metadata (where the name is 'ST_CLASSTYPE' and the value is «TYPE1», «TYPE2», or «TYPE3») displayed by jUCMNav as stereotypes. An intentional element *without* a tag means the related rule applies to all aerodrome types. Using this approach, we eliminate all the elements and dependency links used to define the categories, which reduces the noise and ambiguities while improving the maintainability of the model. We still use resources and dependency links to model

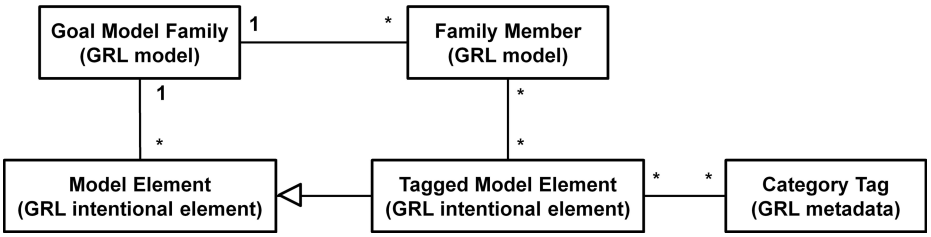

Fig. 3. Conceptual model for our GRL profile for goal model families

conditions that apply to an intentional element. We tag the conditions (metadata named 'ST_CONDITIONTYPE' to differentiate it from the one used for intentional elements, but the values are the same as for 'ST_CLASSTYPE') to specify which aerodrome categories the conditions apply to.

Furthermore, we tag the GRL strategies with metadata (where the name is 'acceptStereotype' and the value is «TYPE1», «TYPE2», or «TYPE3») to specify the model elements that will be evaluated. For instance, when users run a strategy that is stereotyped as «TYPE1» and «TYPE2», all the elements tagged with «TYPE1» and «TYPE2» will be evaluated and tagged elements without those tags will be ignored. The described evaluation method only works with the new proposed algorithm described in the next section.

To ensure traceability with the source legislative documents, two additional stereotypes are used. We tag goals with reference metadata (where the name is 'RegDocRef' and the value is the name of the regulation in the source legislative document) and hyperlink metadata (where the name is 'Hyperlink' and the value is a URL to a section in the source legislative document). The reference metadata enables modelers to query the URN model to spot elements corresponding to a given part of a legislative document, on a name basis, whereas the link metadata establishes clickable traceability links to online legislative documents. Finally, the metadata with name 'ST_Term' is used to provide further information about the structure of the regulation document. The value is a user-defined string that corresponds to one of the many types of sections (e.g., Part, Sub-part, Section, Rule) and hence reflects the structure of a regulation document.

Table 1 gives a complete summary of the stereotypes related to the measured compliance profile and discussed in this section. The stereotypes 'ST_NO', 'IgnoreNodeInEvaluation', and 'Runtime Contribution' are explained in the next section.

In addition, we defined ten well-formedness constraints formalized in UML's Object Constraint Language (OCL) and checked against the model by jUCMNav [7]. These constraints, specified in Appendix A, are further discussed in section 4.3.

Table 1. Measured Compliance Profile Metadata/Stereotypes

Stereotype Name	**Stereotype Value**	**GRL Element**
ST_CLASSTYPE	User enumeration	Goal
ST_CONDITIONTYPE	User enumeration	Resource
acceptStereotype	User enumeration	Strategy
ST_NO	"No"	Goal
IgnoreNodeInEvaluation	N/A (ignored)	Goal, Resource
Runtime Contribution	Computed contribution value	Contribution Link
ST_Term	User enumeration	Goal
RegDocRef	Reference to source document	Goal
Hyperlink	URL to source document	Goal, Resource

4.2 New Analysis Algorithm

There are three main GRL evaluation algorithms (quantitative, qualitative, and hybrid) that are supported by jUCMNav [7]. More recently, a formula-based quantitative algorithm that supports KPI aggregation has been proposed and prototyped [9]. The quantitative evaluation algorithm also supports KPIs. Hence, we extend the quantitative algorithm to take the defined stereotypes on strategies and intentional elements into consideration during the evaluation process, in order to be able to eliminate the intentional elements that are not of a specific type (i.e., not part of the desired family member), and to distribute the weights of the contribution links of the ignored intentional elements among accepted intentional elements. For instance, assume that goal G gets contributions from X, Y, Z, and W with values 45, 30, 10, and 15 respectively (e.g., see the graph in Fig. 4). The standard quantitative algorithm multiplies the intentional elements' satisfaction values by their contribution weights, sums them up, and then divides the total by 100 to calculate the satisfaction level of the target intentional element. The satisfaction value of goal G is computed from the four other goals through their contribution links:

$$((45 \times 45) + (70 \times 30) + (50 \times 15) + (100 \times 10))/100 = \mathbf{58}$$

Now, assume this is a model family where X is tagged with «TYPE1», Y with «TYPE2», W with «TYPE3», and Z with «TYPE1». If we go with a strategy that restricts the family to members of type TYPE1, then the issue with the conventional algorithm is that the contributions to G sum up to 55 (45 + 10), and hence G's satisfaction value cannot be higher than 55. The actual satisfaction value of goal G with only «TYPE1» elements taken into account is:

$$((45 \times 45) + (0 \times 30) + (0 \times 15) + (100 \times 10))/100 = \mathbf{30}$$

With our modified algorithm however, if the strategy is tagged with «TYPE1», then Y and W will be tagged dynamically (and temporarily) with an

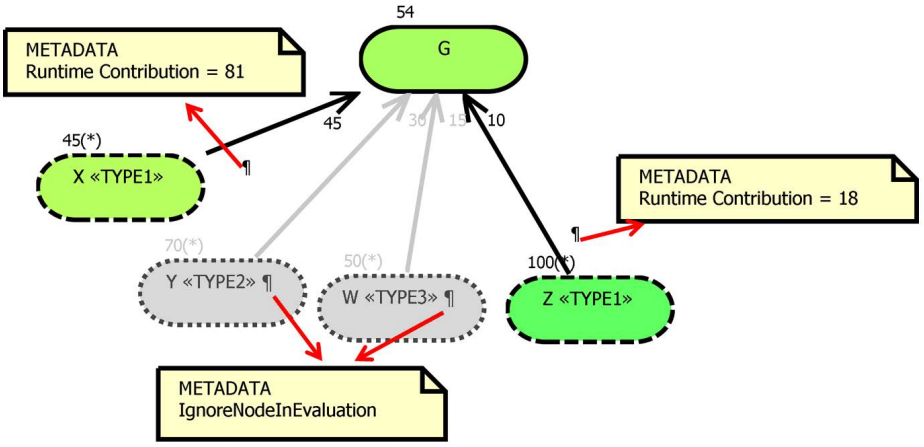

Fig. 4. Example of modified GRL analysis algorithm

'IgnoreNodeInEvaluation' stereotype. The contribution level of ignored elements will be distributed proportionally over the contribution levels of remaining active intentional elements. These elements are tagged with the 'Runtime Contribution' stereotype to show the dynamically calculated contribution values to the modeler. We assume that the sum of contribution levels does not exceed 100; we check this particular style through one of our OCL constraints (see Constraint 1 in Appendix A). In this example, the total contribution value of Y and W computed at analysis time, which is (30+15), is divided proportionally between X and Z as follows:

$$\begin{aligned}
&\text{IgnoredContributionLevel} = 30 + 15 = 45 \quad \textit{// Not of TYPE1: } Y + W\\
&\text{SumConsideredContributionLinks} = 45 + 10 = 55 \quad \textit{// TYPE1: } X + Z\\
&\text{ContributionLevelX} = \text{ContributionLevelX} + (\text{ContributionLevelX} \times\\
&\qquad \text{IgnoredContributionLevel}) \,/\, \text{SumConsideredContributionLinks}\\
&\qquad = 45 + (45 \times 45)/55 = 81\\
&\text{ContributionLevelZ} = \text{ContributionLevelZ} + (\text{ContributionLevelZ} \times\\
&\qquad \text{IgnoredContributionLevel}) \,/\, \text{SumConsideredContributionLinks}\\
&\qquad = 10 + (10 \times 45)/55 = 18
\end{aligned}$$

This leads to a new satisfaction value of **54** for goal G:

$$((45 \times 81) + (100 \times 18))/100 = \mathbf{54}$$

jUCMNav supports this new analysis algorithm and also greys out intentional elements and links that are not part of the selected family member, as shown in Fig. 4. Furthermore, the 'ST_NO' stereotype can be used for optimization and for dropping a model element from the analysis when it is not measurable.

In our family example for aerodromes, Fig. 5 illustrates a strategy-based analysis of perimeter security using our new GRL algorithm. The strategy used here

has been tagged with «TYPE1», therefore only intentional elements and conditions tagged with «TYPE1» are considered during the evaluation. The contribution values from Fence rule1 and Fence rule2 are added to the contribution from Fence rule3, which becomes 99 at analysis time (and the sole contributor to Rules regarding fences).

This new set of features enables the analyst to think of contributions more in terms of relative weights rather than absolute weights. Overall, the measured compliance profile, its adapted propagation algorithm, and the new jUCMNav visualization features help reduce noise in the models (through minimal usage of conditions and stereotypes), handle scalability and maintenance (with proper visualization and a reduced number of symbols for handling multiple models in one family model), resolve the ambiguity between the two types of conditions discussed earlier, and solved evaluation and analysis issues related to the redistribution of contribution weights.

4.3 OCL Well-Formedness Constraints

To ensure the quality of input models before analysts start their analysis, well-formedness constraints must be defined and checked. A set of well-formedness constraints, written in OCL and supported by jUCMNav [6], is presented in Appendix A. These constraints are part of the profile for measured compliance. For example, Constraint 2 will ensure that a GRL resource with metadata 'ST_CONDITIONTYPE' must be a dependee of an intentional element.

Moreover, another OCL constraint (Constraint 9) functions as a query that, on a name basis, finds all goals in the GRL model that represent a specific part of the source legislative document. These constraints take advantage of an OCL library of over 120 pre-defined functions used to query and check URN models, hence simplifying the definition of profile constraints.

Violations to any constraints are reported in jUCMNav's Problems view as a list of problems. By clicking on a problem in the Problems view, the violating element or diagram of the model is highlighted. Figure 6 shows a small example where the ten constraints from the measured compliance profile have been checked and violated. Note that four of these constraints report warnings only as these are either not severe enough to affect the results of the profile's propagation algorithm, or they are simply the result of queries.

5 Related Work and Discussion

There has been much effort devoted to the modeling of variability on goal models. Ali *et al.* [10] present contextual goal models that extend Tropos with variation points where the context may influence the choice among alternatives. They also use tagging with conditions on goals and links (decomposition, dependency, contribution). However, they are more interested in describing runtime adaptation based on a logic-based representation of goals and conditions than in capturing families of related goal models with the automatic adjustment of quantitative contributions. They do not have graphical tool support.

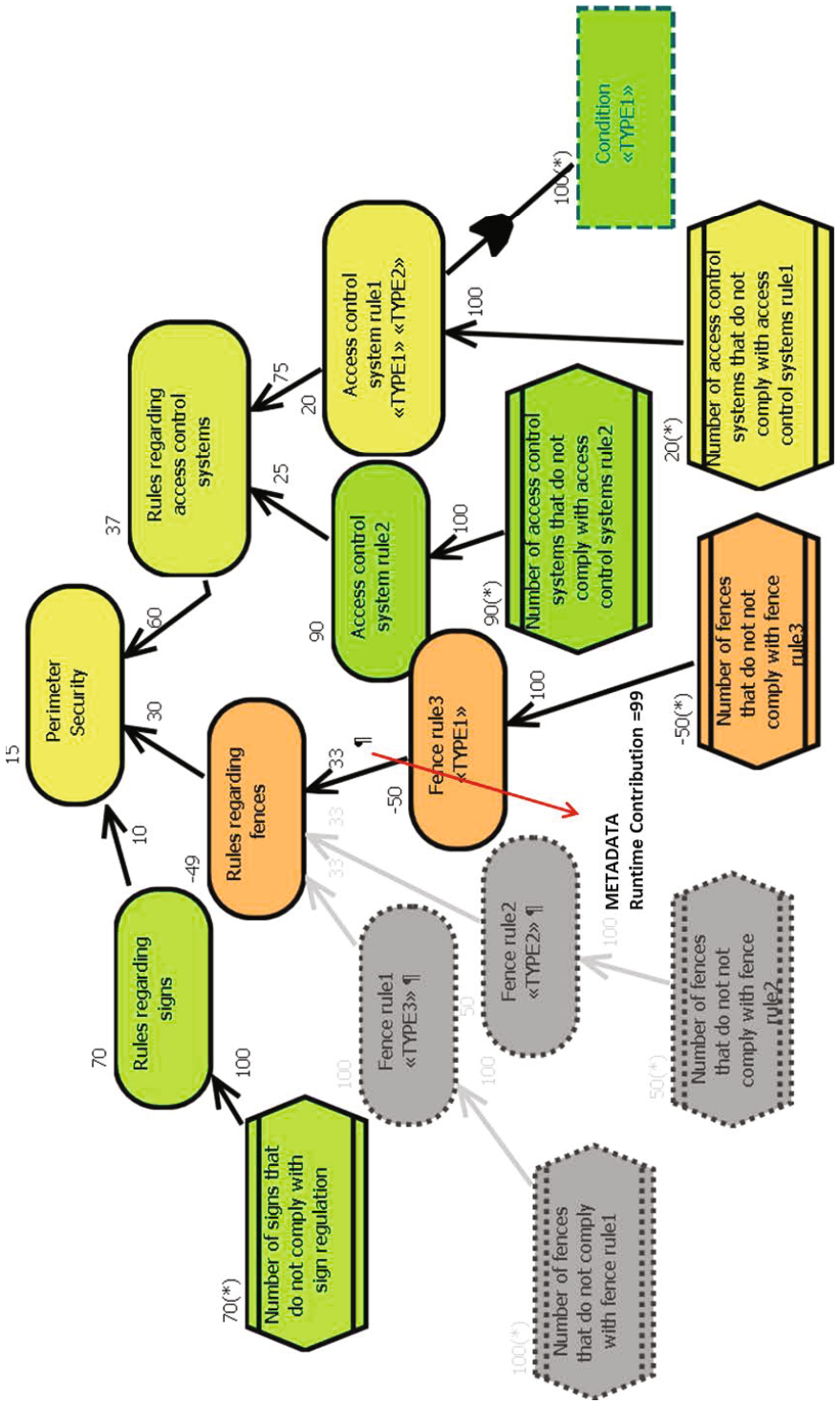

Fig. 5. Example of TYPE1 aerodrome with the modified GRL analysis algorithm

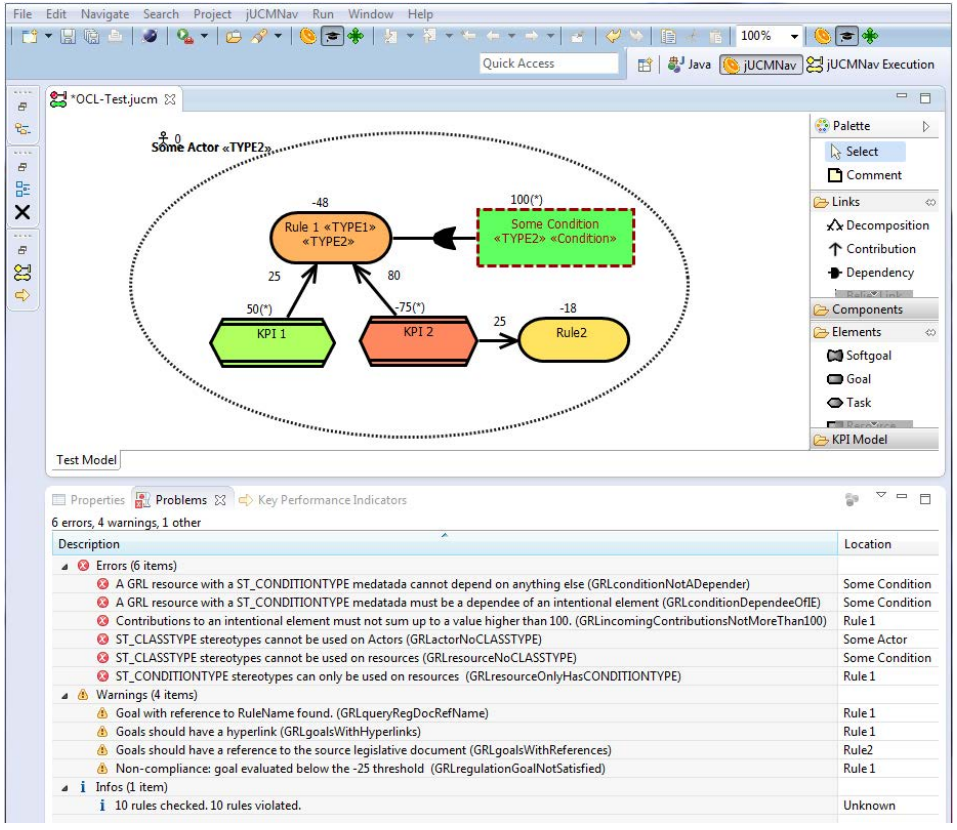

Fig. 6. Example of violations of the profile's OCL constraints in jUCMNav

Lapouchnian *et al.* [11] propose a framework for modeling and analyzing domain variability for goal models. They label model elements that need to be visible with contextual tags. A Boolean variable is assigned to each tag, identifying whether a tag should be active or not. They also propose an algorithm for extracting the parts of the goal model that are dependent on the context variability. Furthermore, they extend the *i** notation to represent and support variations in goal models [12]. Variations of a goal model can be generated from a single context-parameterized *i** model based on the current active contexts. Our approach is similar as we also tag elements with appropriate context information and extract members. However, we take this concept further and use it for quantitative evaluation of the goal models (and not just Boolean evaluation) while automatically adjusting contribution links to produce valid results. In addition in our approach, the tags are visualized with stereotypes, conditions with special intentional elements, and non-member elements with grey shading.

Goal-oriented languages have also been used to support feature models for software product line (SPL). Borba *et al.* [13] provide a comparison between existing goal-oriented techniques for feature modeling in SPL. In particular, Silva *et al.* [14] propose an extension to *i** that enables modeling common and variable features of SPL with cardinalities using tasks and resources of a goal model to capture features. Yu *et al.* [15] propose a tool-based method to create feature models from a goal model. Mussbacher *et al.* [16] review the literature on goal modeling and SPL and propose an SPL framework based on Aspect-oriented URN that allows capturing features and reasoning about stakeholders' needs. Goal models are often used in the literature to express the tradeoffs about non-functional aspects when selecting particular configurations of software products. However, they are not really about families of goal models. To our knowledge, the concept of family of goal models has not been discussed in the literature, and certainly not in a legal compliance context.

A recent systematic literature review revealed that goal-oriented languages have been used to model regulations and compliance [1]. Furthermore, according to another review, some approaches allow organizations to measure their level of compliance [2]. However, none of the existing approaches proposed a technique to analyze individual goal models that are members of a larger family of models.

Although using existing GRL constructs to model conditions addressed our basic needs, this approach can be confusing for current users of GRL models in other application domains. We tried to mitigate this problem by using a different symbol color in the proposed GRL profile (see Condition in Fig. 5), in a way that is compatible with the URN standard. The analysis time contribution information is currently only accessible as metadata in the tool implementation (requiring mouse hovering to be visible), and this might also be confusing to the user.

This profile does not solve the issue of how to choose appropriate contribution levels in goal models. However, this is somewhat mitigated by the availability of recent features in jUCMNav such as *contribution overrides* (with which one can store alternative weights for contributions and use them in combination

with GRL strategies during analysis) and *sensitivity analysis* (through which intervals rather than simple values can be attached to contributions and initial satisfaction levels, and then used with jUCMNav's analysis algorithms) [17].

The tool-supported approach from section 4 has been used on real regulations for aviation security, with models containing hundreds of elements. We have not observed any scalability issue at the moment, but there is no guarantee this approach will fit other regulations or other domains outside compliance, and validation experiments are needed to address these issues. The concept of model families could be adapted to other goal-modeling languages, but in general other languages do not provide all the facilities provided for free by the URN language (e.g., strategies and extension mechanisms) and by the jUCMNav tool, hence porting these ideas might prove to be difficult.

6 Conclusions

In this paper, based on the challenges observed through the modeling of aerodrome security regulations with GRL and key performance indicators, the concept of families of goal models was defined. This led to the creation of a GRL profile for measured compliance, comprised of a set of stereotypes and OCL well-formedness constraints exploited by a novel propagation algorithm, with support for modeling, analysis, and visualization provided by the jUCMNav tool.

Although regulators define regulations (e.g., aviation security) that apply to all the organizations they regulate (e.g., aerodromes), not all rules are usually applicable to all types of organizations. This is a common situation that is not limited to aviation security. We used a perimeter security example to illustrate that when regulations apply to various types of organizations, modeling them using existing goal-modeling approaches can lead to various issues including model noise, scalability, and maintenance problems, ambiguity with how conditions are represented, and misleading evaluation and analysis results. We characterized the requirements for potential solutions and proposed a solution that allows modelers to define a generic family goal model for all types of organizations and tag the model elements to specify which ones are applicable to which family member. The solution is formalized as a profile for GRL, with suitable stereotypes, well-formedness constraints, and an analysis algorithm that exploits this new information. After illustrating and discussing the approach, several limitations and items for future work have been identified. Families of goal models fit nicely the problems observed in regulatory compliance, but we suspect they can address a much larger class of problem domains. Generality and scalability will be explored further in future work.

Acknowledgments. This research was supported by NSERC's Business Intelligence Network. We also thank Andrew Miga for his useful contributions to the jUCMNav tool regarding the new compliance profile.

In memoriam Rasha Tawhid. Sadly, Dr. Tawhid (Ph.D., Carleton University, 2012) passed away in October 2012, following her courageous fight against cancer. She is deeply missed by her family, friends and colleagues.

References

1. Ghanavati, S., Amyot, D., Peyton, L.: A Systematic Review of Goal-oriented Requirements Management Frameworks for Business Process Compliance. In: 4th Int. Work. on Requirements Engineering and Law, RELAW, pp. 25–34. IEEE Computer Society (2011)
2. Shamsaei, A., Amyot, D., Pourshahid, A.: A Systematic Review of Compliance Measurement Based on Goals and Indicators. In: Salinesi, C., Pastor, O. (eds.) CAiSE 2011 Workshops. LNBIP, vol. 83, pp. 228–237. Springer, Heidelberg (2011)
3. Amyot, D., Mussbacher, G.: User Requirements Notation – The First Ten Years, The Next Ten Years. Journal of Software, JSW 6(5), 747–768 (2011)
4. International Telecommunication Union: Recommendation Z.151 (10/12), User Requirements Notation (URN) - Language definition, http://www.itu.int/rec/T-REC-Z.151/en
5. Pourshahid, A., Amyot, D., Peyton, L., Ghanavati, S., Chen, P., Weiss, M., Foster, A.: Business process management with the User Requirements Notation. Electronic Commerce Research, ECR 9(4), 269–316 (2009)
6. jUCMNav, Version 5.2.0, University of Ottawa, http://softwareengineering.ca/jucmnav
7. Amyot, D., Ghanavati, S., Horkoff, J., Mussbacher, G., Peyton, L., Yu, E.: Evaluating Goal Models within the Goal-oriented Requirement Language. International Journal of Intelligent Systems 25(8), 841–877 (2010)
8. Canada Gazette - Government Notices, http://canadagazette.gc.ca/rp-pr/p1/2011/2011-12-10/html/notice-avis-eng.html#d101
9. Pourshahid, A., Richards, G., Amyot, D.: Toward a Goal-Oriented, Business Intelligence Decision-Making Framework. In: Babin, G., Stanoevska-Slabeva, K., Kropf, P. (eds.) MCETECH 2011. LNBIP, vol. 78, pp. 100–115. Springer, Heidelberg (2011)
10. Ali, R., Dalpiaz, F., Giorgini, P.: A Goal-based Framework for Contextual Requirements Modeling and Analysis. Requirements Engineering Journal 15, 439–458 (2010)
11. Lapouchnian, A., Mylopoulos, J.: Modeling Domain Variability in Requirements Engineering with Contexts. In: Laender, A.H.F., Castano, S., Dayal, U., Casati, F., de Oliveira, J.P.M. (eds.) ER 2009. LNCS, vol. 5829, pp. 115–130. Springer, Heidelberg (2009)
12. Lapouchnian, A., Mylopoulos, J.: Capturing Contextual Variability in i* Models. In: Proc. iStar 2011, vol. 766, pp. 96–101. CEUR-WS.org (2011), http://ceur-ws.org/Vol-766/paper17.pdf
13. Borba, C., Silva, C.: A Comparison of Goal-Oriented Approaches to Model Software Product Lines Variability. In: Heuser, C.A., Pernul, G. (eds.) ER 2009. LNCS, vol. 5833, pp. 244–253. Springer, Heidelberg (2009)

14. Silva, C., Borba, C., Castro, J.: A goal oriented approach to identify and configure feature models for software product lines. In: 14th Workshop on Requirements Engineering, WER 2011 (2011)
15. Yu, Y., Leite, J.C.S.P., Lapouchnian, A., Mylopoulos, J.: Configuring features with stakeholder goals. In: Symposium on Applied Computing, SAC 2008, pp. 645–649. ACM Press (2008)
16. Mussbacher, G., Araújo, J., Moreira, A., Amyot, D.: AoURN-based Modeling and Analysis of Software Product Lines. Software Quality Journal 20(3-4), 645–687 (2012)
17. Amyot, D., Shamsaei, A., Kealey, J., Tremblay, E., Miga, A., Mussbacher, G., Alhaj, M., Tawhid, R., Braun, E., Cartwright, N.: Towards Advanced Goal Model Analysis with jUCMNav. In: Castano, S., Vassiliadis, P., Lakshmanan, L.V.S., Lee, M.L. (eds.) ER 2012 2012 Workshops. LNCS, vol. 7518, pp. 201–210. Springer, Heidelberg (2012)

Appendix A Well-Formedness Constraints

This appendix formalizes in OCL the well-formedness constraints of the URN measured compliance profile introduced in section 4. These constraints are available in the jUCMNav tool and make use of jUCMNav's library of predefined OCL functions. Note that Constraint 9 functions as a query and that Constraint 10 is meant to be checked on a model where a GRL strategy is evaluated.

Constraint 1. Contributions to an intentional element must not sum up to a value higher than 100 (Reason: ensures that *all* contributing elements must evaluate to 100 for this intentional element to be fully satisfied).

```
context grl::IntentionalElement
inv GRLincomingContributionsNotMoreThan100:
  self.linksDest
    -> select(link | link.oclIsTypeOf(grl::Contribution))
    -> collect(link | link.oclAsType(grl::Contribution))
      .quantitativeContribution
    -> sum() <= 100
```

Constraint 2. A GRL resource with a ST_CONDITIONTYPE metadata must be a dependee of an intentional element (Reason: ensures that conditions are used as specified by the measured compliance profile). Note: the jUCMNav metamodel uses the term Ressource rather than Resource.

```
context grl::IntentionalElement
inv GRLconditionDependeeOfIE:
  (self.type=IntentionalElementType::Ressource and
    self.hasMetadata('ST_CONDITIONTYPE'))
  implies
  self.linksSrc
    -> select(link | link.oclIsTypeOf(grl::Dependency))
    -> collect(link | link.oclAsType(grl::Dependency)).dest
    -> select(le | le.oclIsTypeOf(grl::IntentionalElement))
    -> size() > 0
```

Constraint 3. ST_CLASSTYPE stereotypes cannot be used on actors (Reason: ensures that this stereotype is used as specified by the measured compliance profile).

```
context grl::Actor
inv GRLactorNoCLASSTYPE:
  not(self.hasMetadata('ST_CLASSTYPE'))
```

Constraint 4. A GRL resource with a ST_CONDITIONTYPE stereotype cannot depend on anything else (Reason: ensures that conditions are used as specified by the measured compliance profile).

```
context grl::IntentionalElement
inv GRLconditionNotADepender:
  (self.type=IntentionalElementType::Ressource and
    self.hasMetadata('ST_CONDITIONTYPE'))
  implies
  self.linksDest
    -> select(link | link.oclIsTypeOf(grl::Dependency))
    -> isEmpty()
```

Constraint 5. ST_CONDITIONTYPE stereotypes can only be used on resources (Reason: ensures that stereotypes are used as specified by the measured compliance profile).

```
context grl::IntentionalElement
inv GRLresourceOnlyHasCONDITIONTYPE:
  not((self.type=IntentionalElementType::Ressource))
  implies
  not(self.hasMetadata('ST_CONDITIONTYPE'))
```

Constraint 6. ST_CLASSTYPE stereotypes cannot be used on resources (Reason: ensures that stereotypes are used as specified by the measured compliance profile).

```
context ggrl::IntentionalElement
inv GRLresourceNoCLASSTYPE:
  (self.type=IntentionalElementType::Ressource)
  implies
  not(self.hasMetadata('ST_CLASSTYPE'))
```

Constraint 7. Goals should have a hyperlink (violations shown as warnings only) (Reason: ensures that the source legislative documents are accessible from the GRL model). Note: this does not ensure that the specified hyperlink is valid.

```
context grl::IntentionalElement
inv GRLgoalsWithHyperlinks:
  self.type=IntentionalElementType::Goal
  implies
  self.hasMetadata('hyperlink')
```

Constraint 8. Goals should have a reference to the source legislative document (violations shown as warnings only) (Reason: ensures that there is a name that can be queried by Constraint 9). Note: this does not ensure that the specified name is valid.

```
context grl::IntentionalElement
inv GRLgoalsWithReferences:
  self.type=IntentionalElementType::Goal
  implies
  self.hasMetadata('RegDocRef')
```

Constraint 9. Functions as a query: finds goals where the reference metadata equals RuleName (shown as warnings only). Note: RuleName is a parameter here and can be substituted with any name. (Reason: ensures that the GRL model can be queried for elements traceable from a specific part of a legislative document as specified by RuleName.)

```
context grl::IntentionalElementRef
inv GRLqueryRegDocRefName:
  not(getDef().getMetadata('RegDocRef') = 'RuleName'
    and
    getDef().type=IntentionalElementType::Goal)
```

Constraint 10. Non-compliance: goal evaluated below the -25 threshold (violations shown as warnings only). Note: the -25 value is a default threshold but could be set to a different value by the analyst. (Reason: this constraint highlights regulation goals against which the organization performs poorly.)

```
context grl::IntentionalElementRef
inv GRLregulationGoalNotSatisfied:
  getDef().type=IntentionalElementType::Goal
  implies
  getDef().getNumEval() > -25
```
