RationalGRL: A Framework for Rationalizing Goal Models Using Argument Diagrams

Marc van Zee[1], Diana Marosin[2], Floris Bex[3], and Sepideh Ghanavati[4]

[1] Computer Science and Communication, University of Luxembourg, Luxembourg
[2] Luxembourg Institute of Science and Technology, Luxembourg
[3] Information and Computing Sciences, Utrecht University, The Netherlands
[4] ICIS, Radboud University, The Netherlands
marc.vanzee@uni.lu, diana.marosin@list.lu, s.ghanavati@cs.ru.nl, f.j.bex@uu.nl

Abstract. Goal modeling languages, such as i^* and the Goal-oriented Requirements Language (GRL), capture and analyze high-level goals and their relationships with lower level goals and tasks. However, in such models, the rationalization behind these goals and tasks and the selection of alternatives are usually left implicit. To better integrate goal models and their rationalization, we develop the RationalGRL framework, in which argument diagrams can be mapped to goal models. Moreover, we integrate the result of the evaluation of arguments and their counter-arguments with GRL initial satisfaction values. We develop an interface between the argument web tools OVA and TOAST and the Eclipse-based tool for GRL called jUCMNav. We demonstrate our methodology with a case study from the Schiphol Group.

Key words: Goal modeling languages, decision rationalization, argumentation theory, argument diagrams

1 Introduction

The Goal-oriented Requirements Language (GRL) is part of the User Requirements Notation (URN), which is an ITU-T standard [8]. GRL [1] consists of several intentional elements (such as softgoals, goals, tasks and resources) and links between them. While GRL models, and goal models in general [19], are useful tools for motivating architectural choices in enterprise and software architecture, they miss important parts of the architecture design rationalization. As a result, GRL models are only the end product of a modeling process, and they do not provide any insight on how the models were created, i.e., what reasons were used to choose certain elements in the model and to reject the others and what evidence was given as the basis of this reasoning.

In this paper, we integrate various existing and newly developed interfaces and algorithms into the RationalGRL framework. This framework facilitates argument construction and analysis on the one hand and the rationalization and

evaluation of goal models on the other hand. More specifically, RationalGRL framework combines an existing argument diagramming tool[5] [4] based on a formal theory of practical (i.e. goal-driven) argumentation [2, 10, 16, 17] with a standardized goal modeling language and its tool support [1].

The core of RationalGRL is a concrete set of mapping rules from the formal argumentation framework to a GRL model. The mapping rules allow for the automatic translation of arguments and evidence about goals to GRL models. Furthermore, the formal semantics [5] of arguments and counterarguments underlying the argumentation theory helps determining whether the elements of a GRL model are acceptable given the potential contradictory evidence and stakeholders' opinions. In other words, we can compute the initial satisfaction level of IEs in GRL based on the acceptability status of arguments for or against IEs. RationalGRL framework is implemented as an online tool [6].

The rest of this paper is structured as follows. In Section 2, we briefly introduce the RationalGRL framework. Due to space constraints we omit technical details. In Section 3, we evaluate the framework via a case study of the Schiphol Group. First, we model the discussions about a change in the set of architecture principles of the Schiphol Group. Second, we evaluate the framework and the resulted models with enterprise architects of the Group. In Section 4 we present the current literature and the related work.

2 The RationalGRL Framework

The main components of the RationalGRL framework are shown in Figure 1. The four main parts of the framework, Argumentation, Translation, Goal Modeling, and Update, are numbered and depicted in **bold**. For each component, the technology used to implement it, is marked in a filled rectangle. The last step (*Update*) is out of the scope of this paper. We, now, briefly explain the process of how a goal model is developed in the RationalGRL framework.

In *Step 1 - Argumentation*, stakeholders discuss the requirements of their organization. In this process, stakeholders put forward arguments for or against certain elements of the model (e.g. goals, tasks,..). Arguments about why certain tasks can contribute to the fulfillment of goals and an evidence to support a claim are also part of this process. Furthermore, stakeholders can challenge claims by forming counterarguments. The complete set of claims, arguments and counterarguments can be represented in an argument diagram.

In *Step 2 - Translation*, the argument diagram is translated to a goal model, in our case GRL. In addition to the structure of arguments and counterarguments, this step also provides means to translate the evaluation of arguments in the argument diagrams to the initial satisfaction values of the GRL intentional elements, which can be positive or negative. A positive evaluation indicates that

[5] `http://ova.arg-tech.org/`

[6] All implementation details/sources as well as the case study descriptions and models can be found on Github: `http://github.com/RationalArchitecture/RationalGRL`.

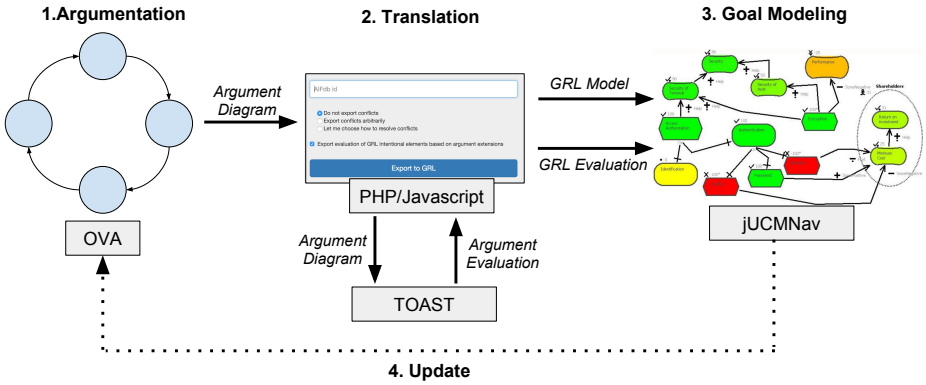

Fig. 1: Overview of the RationalGRL Framework

the element is supported by one or more arguments, while a negative evaluation indicates that the element is not a good alternative in GRL.

In *Step 3 - Goal Modeling*, the goal model that is generated by the Translation process, is evaluated by the stakeholders. These models can be used as a discussion means to investigate whether the goals in the model are in line with the original requirements of the stakeholders. This allows a better rationalization of the goal modeling process, with a clear traceability from the goals of the organization to the arguments and evidence that were used in the discussions.

Step 4 - Update involves translating GRL models with its analysis back into an argument diagram. This falls outside the scope of the current paper.

3 Evaluation

3.1 Case Study Description

Schiphol Group is the owner of an international airport and it operates on both national and international scales. Schiphol Group started using enterprise architecture principles to drive their architecture program in 2003. Principles are generally defined as "a family of guidelines (...) for design" [7] or "general rules and guidelines, intended to be enduring and seldom amended, that inform and support the way an organization fulfils its mission" [14]. In 2003, the principle Adhere to the Corporate Data Model was advocating the use of a company-wide defined data model, such that it provided a high level insight on all the data that were used in the processes and applications. In 2007, the principles were evaluated by a team of five architects and it was concluded that the principle was not very successful, and was conceptually conflicting with another architecture principle Package selection before custom development.

We use these principles as the base of our case study and we provide *a posteriori* analysis of the discussions and evidence that were used in forming the

architecture principles and present the goal models generated using the RationalGRL framework. In total, we formalized around 60 arguments and 30 inferences/attack relation. Due to space constraints, we only provide a small subset of the models in this paper.

3.2 Modeling the Case Study in RationalGRL

In 2003, the principle Adhere to the Corporate Data Model was advocating for the use of a company-wide defined data model, such that it provided a high level insight on all the data that is used in the processes and applications. The IT department was assigned the task to define this data model. All applications were supposed to be compatible to this data model. The main motivation for adopting this principle was to obtain a clear and standard approach for information handling. This could improve the way in which customers can be served and to lower the costs.

In 2007, the principles were re-evaluated by a team of five architects. It was concluded that the previous principle was not very successful. Some of the arguments used in this discussion are shown in Figure 2. An important issue with the corporate data model - principle was the effort needed to be invested by the ICT department to define this data model. Schiphol Group not only focuses on aviation, but also on retail and security. These domains have different needs when it comes to the data they use and their internal processes. In terms of business objects and "on paper" definitions, the data model was agreed upon, but it was never really implemented.

This situation is reflected in a simplified way in the argument diagram by an attack from the argument [EVIDENCE] CorpDM is not defined on [TASK] ICT department defines CorpDM. Two of the other attacks are direct consequences of this issue. Since there was no corporate data model, the principle could not be used ([EVIDENCE] Principle has been used minimally... use it now), and databases between applications were seldom shared ([EVIDENCE] Databases seldom shared).

In addition, the principle was conceptually conflicting with another architecture principle Package selection before custom development. It was virtually impossible to find third-party applications and vendor packages that comply with the corporate data model. This is reflected in the argument diagram as a bi-directional conflict between the principle and [TASK] Use data models of packages applications instead of CorpDM. This task is a direct consequence of the principle Package selection before custom development.

This set of arguments and the evaluation of the real-life situation made architects realize that the focus should be on the exchange of information between applications, not on how the data is stored and managed centrally. This shift of paradigm resulted in creating a new principle Adhere to the canonical data model.

We translate the final argument diagram of the situation in 2007 after introducing the new principle using RationalGRL framework. The result is presented in Figure 3. Based on the evaluation of GRL IEs, the new principle as well as goals such as Lower diversity and total cost of ownership, Clear and standard way of interfacing, Few dependencies between applications, and Faster time to market receive

a positive evaluation. Moreover, the old principle receives a negative evaluation, together with its related goals and tasks. This conclusions can provide insights on how to prioritize from a set of principles, or how to take better informed design decisions when facing alternative solutions.

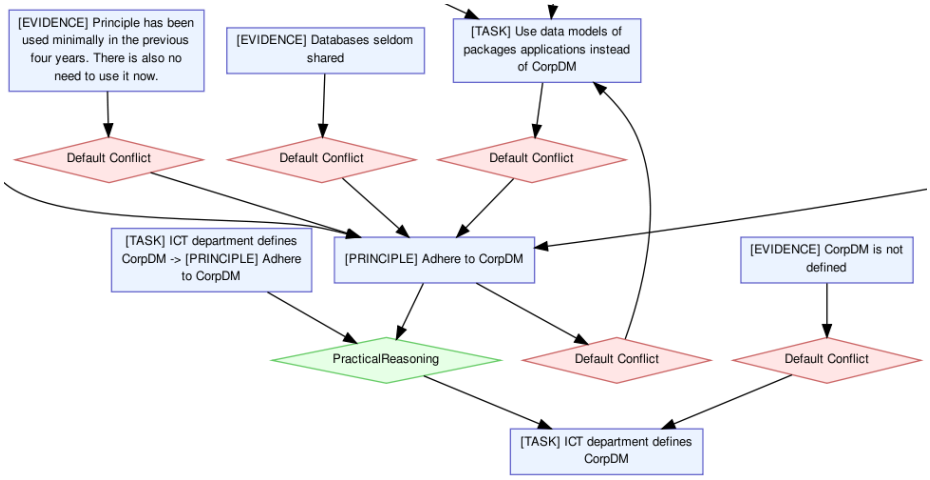

Fig. 2: Part of the argument diagram of the Schiphol Group principles 2007 (Visualized in the Argument Web)

3.3 Evaluation with Schiphol Group Enterprise Architects

We evaluated our framework and its results with enterprise architects of the Schiphol Group. We first discussed the argument diagrams in order to evaluate whether they represent the situation at 2003 and 2007 correctly. The architects found that argument diagrams are a useful tool to link and reason about arguments. However, they noted that it may be easier to construct the arguments and counterarguments *a postiori* than to do this *a priori*. They felt that it is easier to look back on the process and to extract that relevant arguments, than to do this while the process is still ongoing.

Next, we translated the argument diagrams to GRL models using the translation procedure and evaluated these GRL models with the architects. The architects confirmed that the models are able to represent correctly part of the problem at hand. However, they also noted that some parts were missing from the models, which implies that beside the documents we gathered and used for modeling, there were additional facts we did not consider. However, this partial representation was found useful and the architects consider the usage of formal methods (such as GRL and argumentation) beneficial for "sanity checks", alongside a better formulation of the principles.

Fig. 3: GRL Diagram of the Schiphol Group principles in 2007

4 Related Work

There are several contributions that relate argumentation-based techniques with goal modeling. The contribution most closely related to ours is the work by Jureta *et al.* [9]. This work proposes "Goal Argumentation Method (GAM)" to guide argumentation and justification of modeling choices during the construction of goal models. One of the elements of GAM is the translation of formal argument models to goal models (similar to ours). In this sense, our RationalGRL framework can be seen as an instantiation and implementation of part of the GAM. One of the main contribution of RationalGRL is that it also takes the acceptability of arguments as determined by the argumentation semantics [5] into account when translating from arguments to goal models. RationalGRL also provides tool support for argumentation, i.e. Argument Web toolset, to which OVA belongs [4], and for goal modeling, i.e. jUCMNav [12]. Finally, RationalGRL is based on the practical reasoning approach of [3], which itself is also a specialization of Dung's [5] abstract approach to argumentation. Thus, the specific critical questions and counterarguments based on these critical question proposed by [3] can easily be incorporated into RationalGRL.

RationalGRL framework is also closely related to frameworks that aim to provide a design rationale (DR) [13], an explicit documentation of the reasons behind decisions made when designing a system or artefact. DR looks at issues, options and arguments for and against the various options in the design of, for example, a software system, and provides direct tool support for building

and analyzing DR graphs. One of the main improvements of RationalGRL over DR approaches is that RationalGRL incorporates the formal semantics for both argument acceptability and goal satisfiability, which allow for a partly automated evaluation of goals and the rationales for these goals.

Arguments and requirements engineering approaches have been combined by, among others, Haley *et al.* [6], who use structured arguments to capture and validate the rationales for security requirements. However, they do not use goal models, and thus, there is no explicit trace from arguments to goals and tasks. Furthermore, like [9], the argumentative part of their work does not include formal semantics for determining the acceptability of arguments, and the proposed frameworks are not actually implemented. Murukannaiah *et al.* [11] propose ArgACH, an approach to capture inconsistencies between stakeholders' beliefs and goals, and resolve goal conflicts using argumentation techniques.

Finally, our in previous empirical work we recognized shortcomings in the current state of the art in EA decision rationalization [15]. One of the main shortcomings is that the group decision process is often omitted. This contribution can be seen as way to meet this shortcoming, and in this sense improves on existing EA decision rationalization frameworks. [18]

5 Conclusions and Future Work

There are many directions of future work. There are a large number of different semantics for formal argumentation, that lead to different arguments being acceptable or not. It would be very interesting to explore the effect of these semantics on goal models. Jureta *et al.* develop a methodology for clarification to address issues such as ambiguity, overgenerality, synonymy, and vagueness in arguments. Atkinson *et al.* [2] define a formal set of critical questions that point to typical ways in which a practical argument can be criticized. We believe that critical questions are the right way to implement Jureta's methodology, and our framework would benefit from it. In addition, currently, we have not considered the *Update* step of our framework (Figure 1). That is, the translation from goal models to argument diagrams is still missing. The *Update* step helps analysts change parts of the goal model and analyze its impact on the underlying argument diagram. Finally, the implementation is currently a browser-based mapping from an existing argument diagramming tool to an existing goal modeling tool. By adding an argumentation component to jUCMNav, the development of goal models can be improved significantly.

References

1. Daniel Amyot. Introduction to the user requirements notation: Learning by example. *Computer Networks*, 42(3):285–301, June 2003.
2. Katie Atkinson and Trevor Bench-Capon. Practical reasoning as presumptive argumentation using action based alternating transition systems. *Artificial Intelligence*, 171(10):855–874, 2007.

3. Katie Atkinson and Trevor Bench-Capon. Taking the long view: Looking ahead in practical reasoning. In *Computational Models of Argument: Proceedings of COMMA*, pages 109 – 120, 2014.
4. Floris Bex, John Lawrence, Mark Snaith, and Chris Reed. Implementing the argument web. *Communications of the ACM*, 56(10):66–73, 2013.
5. Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–358, 1995.
6. Charles B Haley, Jonathan D Moffett, Robin Laney, and Bashar Nuseibeh. Arguing security: Validating security requirements using structured argumentation. In *in Proc. of the Third Symposium on RE for Information Security (SREIS'05)*, 2005.
7. Jan A. P. Hoogervorst. Enterprise architecture: Enabling integration, agility and change. *Int. J. Cooperative Inf. Syst.*, 13(3):213–233, 2004.
8. ITU-T. Recommendation Z.151 (11/08): User Requirements Notation (URN) – Language Definition. `http://www.itu.int/rec/T-REC-Z.151/en`, 2008.
9. I.J. Jureta, S. Faulkner, and P.Y. Schobbens. Clear justification of modeling decisions for goal-oriented requirements engineering. *Requirements Engineering*, 13(2):87–115, May 2008.
10. Sanjay Modgil and Henry Prakken. A general account of argumentation with preferences. *Artificial Intelligence*, 195:361–397, 2013.
11. Pradeep K Murukannaiah, Anup K Kalia, Pankaj R Telangy, and Munindar P Singh. Resolving goal conflicts via argumentation-based analysis of competing hypotheses. In *23rd Int. Requirements Engineering Conf.*, pages 156–165. IEEE, 2015.
12. Gunter Mussbacher and Daniel Amyot. Goal and scenario modeling, analysis, and transformation with jUCMNav. In *ICSE Companion*, pages 431–432, 2009.
13. S. J. Buckingham Shum, A. M. Selvin, M. Sierhuis, J. Conklin, C. B. Haley, and B. Nuseibeh. Hypermedia support for argumentation-based rationale. In *Rationale management in software engineering*, pages 111–132. Springer, 2006.
14. The Open Group. TOGAF 9 - The Open Group Architecture Framework Version 9, 2009.
15. Dirk van der Linden and Marc van Zee. Insights from a Study on Decision Making in Enterprise Architecture. In *PoEM (Short Papers)*, volume 1497 of *CEUR Workshop Proceedings*, pages 21–30, 2015.
16. Marc van Zee, Floris Bex, and Sepideh Ghanavati. Rationalization of Goal Models in GRL using Formal Argumentation. In *Proceedings of RE: Next! track at the Requirements Engineering Conference 2015 (RE'15)*, August 2015.
17. Marc van Zee and Sepideh Ghanavati. Capturing Evidence and Rationales with Requirements Engineering and Argumentation-Based Techniques. In *Proc. of the 26th Benelux Conf. on Artificial Intelligence (BNAIC2014)*, November 2014.
18. Marc Van Zee, Georgios Plataniotis, Dirk van der Linden, and Diana Marosin. Formalizing enterprise architecture decision models using integrity constraints. In *2014 IEEE 16th Conference on Business Informatics*, volume 1, pages 143–150. IEEE, 2014.
19. Eric SK Yu. Towards modelling and reasoning support for early-phase requirements engineering. In *Proc. of the 3rd IEEE Int. Symposium on RE*, pages 226–235, 1997.