Capturing Evidence and Rationales with Requirements Engineering and Argumentation-Based Techniques

Marc van Zee [a] Sepideh Ghanavati [b]

[a] *University of Luxembourg, Luxembourg*
[b] *Public Research Center Henri Tudor, Luxembourg*

Abstract

User Requirements Notation (URN) is a modeling language with two complementary notations for capturing goals and non-functional requirements as well as low level processes and functional requirements. URN helps documenting the rationale behind the decisions, however, it lacks gathering the evidences for these rationales and documenting stakeholder discussions. In this paper, we propose a framework to extend URN with the Hybrid approach in reasoning based on evidences.

1 Introduction

Requirements Engineering (RE) has become an essential part of all development processes, especially in the area of complex reactive and distributed systems. The User Requirements Notation (URN) [11], which is based on ITU-T Standard, specifies visual notations to analyze functional (behaviour) and non-functional requirements (NFRs, such as performance, cost, security, and usability). URN consists of two complementary modeling notations: Goal Requirements Language (GRL) and Use Case Maps (UCMs). GRL aims at capturing business or system goals, alternative decisions to achieves high-level goals, and rationales behind goals and decisions. UCM tries to model functional and behaviorial requirements (requirements defining functions or behaviors of the system under development) using scenarios. A model containing GRL diagrams and UCMs is the result of an iterative refinement from both directions, based on discussions between stakeholders, who naturally give arguments for or against decisions, where these arguments can be motivated by evidence. Although URN offers traceability between GRL elements and UCM elements, it is not possible to trace back elements of the models to the actual discussions between stakeholder and the evidence that this was based on.

In this paper, we propose to extend URN to capture discussions between stakeholders based on evidence. We base our extension on the *hybrid approach* from evidential reasoning about the facts of a criminal case [4], in which arguments and narratives can be used in conjunction as well as interchangeably. This hybrid approach was shown to be natural way of modeling the process of proof, the iterative process of constructing, testing and justifying hypotheses in crime investigation and decision-making [7].

We show that parts of a GRL diagram can be traced back to arguments and evidence. Arguments can attack and support each other, which allowed the designer to reason about the impact of changing argument structures or evidence on the GRL diagrams and UCMs. Moreover, using the hybrid approach we are also able to reason about the plausibility, consistency and completeness of UCMs by asking so-called *critical questions*.

1.1 Related Work

Goal modeling notations such as Goal-oriented Requirements Language (GRL) [3] together with Use Case Maps (UCM) [17] have been used to help decision makers analyze alternatives, capture rationale behind the decisions [18, 13]. Gross and Yu [10] explored the use of goal-oriented approaches to provide links between business goals, architectural design decisions and structure in a systematic way.

These notations evaluate the satisfaction level of the high-level, fuzzy goals of the organization via selecting a set of alternatives at lower levels. These analysis mainly focus on qualitative data range. However, Akhigbe et al. [9] introduce a method to be able to quantify the analysis evaluations for goals models. They propose the use of AHP technique in EA to perform the analysis techniques quantitatively. Furthermore, the Business Intelligence - Enabled Adaptive Enterprise Architecture (BI-EAEA) framework [1] has been proposed to use BI theme to help decision making process in EA as well as to support organizational business objectives alignements to daily operations and to provide a coherent view across the domains of the EA. Issue-Based Information System (IBIS) was invented by Werner Kunz and Horst Rittel as an argumentation-based approach to solving wicked problems, which are complex, ill-defined problems that involve multiple stakeholders [12]. Recently, there has been a renewed interest in IBIS-type systems, particularly in the context of sensemaking and collaborative problem solving in a different social and technical contexts. Our approach is comparable to IBIS, except that we make an explicit connection between arguments and stories, while IBIS mostly focuses on arguments.

The rest of this paper is organised as follows. In Section 2 we introduce our running example. In Section 3, we introduce the User Requirements Notation (URN), together with its sub-notations, GRL and UCM scenario notations, followed by a discussion on existing traceability within the URN. In Section 4, we introduce the formal theory of story schemes and arguments using in reasoning about facts of a crime and we also show how we can model a UCM scenario as a story scheme with arguments, and we demonstrate that it leads to an increase of traceability using our running example.

2 Running Example

The stakeholders of a fictitious company “Best Furniture Inc.” have agreed through several meetings that they should improve customer support. The reason behind this is logical: Improving customer support leads to more profit because it will in the long term result in more customer, and more customers leads to more profit. It is clear that increasing profit is one of the main goal of any company.

In order to implement this goal, they have decided that customers should be able to return a product when they are not satisfied with it. The stakeholders initially agree that the following three conditions that should be satisfied if a customer would like to return a product:

1. The product is bought from company “Best Furniture Inc.”.
2. The customer has a receipt for the product.
3. The product is not damaged.

Condition (1) and (2) are separated due to the rationale that every product of the company “Best Furniture Inc.” is registered in their system, thus they are able to check whether the product is bought without a receipt. The rationale for condition (3) is clear: The company is unable to resell damaged products.

After some more discussion, the stakeholders decide that it would be useful to make a distinction between condition (1) and (2), and they decide that if a customer does not have a receipt, he or she is still entitled to company store credit. The stakeholders believe that this will increase their customer support even more. Their statistics also show that the number of customers trying to return a product without a receipt is relatively low, thus it will not cost them much.

3 The User Requirements Notation (URN)

User Requirements Notation (URN) [11], an ITU-T Standard, is one of the first modeling languages in the area of RE which aims at the standardization of visual notations to analyze functional (behaviour) and non-functional requirements (NFRs) such as performance, cost, security, and usability. URN allows software and requirements engineers to identify requirements for a proposed or an evolving system and to review such requirements for correctness and completeness. URN combines two complementary notations: the *Goal-oriented Requirement Language* (GRL) [3] for goals and NFRs, and *Use Case Maps* (UCMs) [17] for scenarios, business processes and functional requirements. GRL, which is based on some existing goal modeling notations (i.e. *i** and NFR Framework), captures business or system goals, alternative ways of satisfying goals and rationales for alternatives, their impact and the decisions. UCM helps modeling functional requirements, business processes and architecture structures using scenarios.

URN supports the reasoning about scenarios by establishing links between intentional elements (such as softgoals and goals) in GRL and non-intentional elements in UCM. Modeling both goals and scenarios is complementary and may aid in identifying further goals and additional scenarios (and scenario steps) important to stakeholders, thus contributing to the completeness and accuracy of the requirements.

The Use Case Map (UCMs) scenario notation focuses on causal relationships between activities or processes of one or more use cases. The relationships are said to be causal due to their involvement in concurrency and partial orderings of responsibilities, their links between causes and effects, and the level of abstraction from component interactions expressed as message exchanges. UCMs are applicable to use case capturing and elicitation, use case validation, as well as high-level architectural design and test case generation. They provide a behavioural framework for evaluating and making architectural decisions at a high level of design.

URN has an open source Eclipse-based tool support called jUCMNav [14]. jUCMNav supports modeling UCM, GRL and the *URN links*(▶) between them. jUCMNav provides the support for multiple view of the same model to help scalability. The tool prevents the creation of syntactically incorrect URN models through hard-coded rules, and more specific styles can be enforced through user-selectable semantic rules written in Object Constraint Language. jUCMNav supports several GRL evaluation algorithms such as quantitative, qualitative, hybrid, constraint-based or conditional evaluation algorithms. However, there is only one UCM path traversal scenario mechanism (with a few parameters). It also allows exporting individual or complete models to different graphical and/or report formats and exporting the results of GRL strategy evaluations in a comma-separated value (CSV) files.

3.1 Basics of GRL Notation

GRL syntax is based on *i** language which describes business concerns such as low cost, fast time to market, high customer satisfaction as well stakeholders' beliefs and dependencies. *Softgoals*( ), are goals which are somewhat fuzzy in nature and cannot be entirely satisfied. *Softgoals* capture high-level objectives. *Goals*( ) represent goals that are not fuzzy and can be clearly achieved. *Tasks* ( ), are typically used to operationalize parent (soft-)goals and/or goals. Goals, softgoals and tasks can be connected to each other, in an AND/OR/XOR graph, using *decomposition* links(+—), or can be connected with *contribution*(→) or *correlation*(⇢) links showing the impact on each other. Contribution or correlation links may have various degrees of impact qualitatively, including necessary and sufficient (*make*), necessary but insufficient (*help*), unknown positive (*some+*), and their corresponding contributions on the negative side (*break, hurt*, and *some-*). Contribution and correlation links have quantitative counterparts, which range from -100 to 100. Justifications or explanations of GRL intentional elements and links can be added in the form of *beliefs* ( ). Beliefs document the rationale behind various parts of a GRL model. Stakeholders can support or argue against such beliefs while validating the model. Beliefs provide a means to prevent stakeholders from having the same discussion over and over again. Although we will only discuss this simple example due to space constraints, GRL models contain much more components. See [2] for a more detailed overview.

Consider the GRL diagram in Figure 1, which is the resulting goal model of our running example. Both the goal Return Product for Money and Return Product for Company Credit contribute positively to the softgoal Improve Customer Support. The goal Return Product is decomposed into three subgoals Bought Item, Have Receipt, and Not Damaged. These subgoals are materialized with the tasks Check if the Item is Bought, Check for Receipt, and Check for Damage. Similarly, the goal Return Products for Company Credit has two subgoals which are materialized by two tasks. The rationalization of the different elements is provided by attaching belief elements to corresponding GRL elements.

3.2 Basic UCM Notation

In UCMs, a *scenario* is a partial description of the system usage defined as a set of partially-ordered responsibilities a system performs to transform inputs to outputs while satisfying preconditions and post-

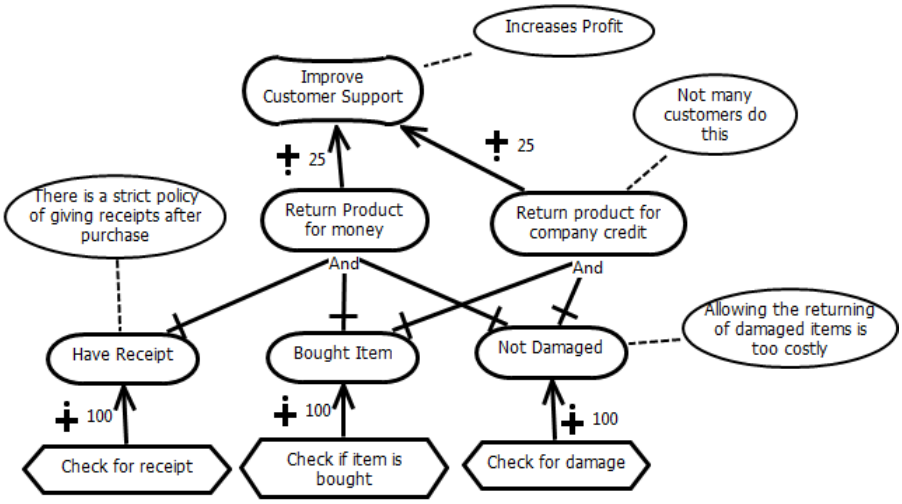

Figure 1: GRL Model for Customer Support

conditions. UCM *responsibilities*(×) are scenario activities representing something to be performed or some tasks to be achieved. A responsibility can potentially be associated or allocated to a *component*(▭). In UCMs, a component is generic and abstract enough to represent software entities (e.g. objects, processes, databases, or servers) as well as non-software entities (e.g. actors, agents or hardware).

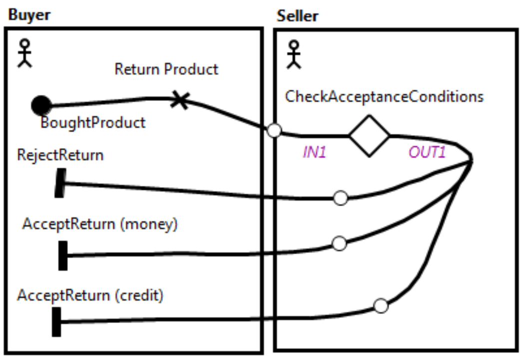

Figure 2: UCM Model for Product Return Procedure

The UCM in Fig 2 illustrates some of these concepts through a description of the product return procedure from our running example. Filled circles represent *start points*(●), which capture preconditions and triggering events. *End points*(▮) capturing resulting events and postconditions are illustrated with bars perpendicular to causal *paths*. Responsibilities can be assigned to *components* such as agents. In this example, there are two agents, namely the buyer and the seller. Scenarios progress along paths from start points to end points. Paths also include responsibilities. Paths can fork as alternatives (*OR-fork*(⌐)) and may also join (*OR-join*) or they can show concurrency with *AND-fork* (⊣⊢) and *AND-join*. Alternative branches can be guarded by *conditions*, shown between square brackets. A condition needs to be true for the guarded path to be followed.

UCM also contains a diamond symbol called a *stub*(◇), which is used as a container for a sub-processes. The stub can be dynamic or static. Dynamic stubs have more than one sub-maps whereas static stubs have only one sub-map. These sub-processes are called *plug-in* maps (since they are plugged into a stub). The plug-in map for the stub CheckAcceptanceConditions is depicted in Figure 3. Stubs have identifiable input and output segments $(IN1, OUT1, \ldots)$ connected to start points and end points in the plug-in. This *binding relationship* ensures that paths flow from parent maps to sub-maps, and back to parent maps.

In this example, the return conditions are twice represented as an OR-fork, allowing the possibility

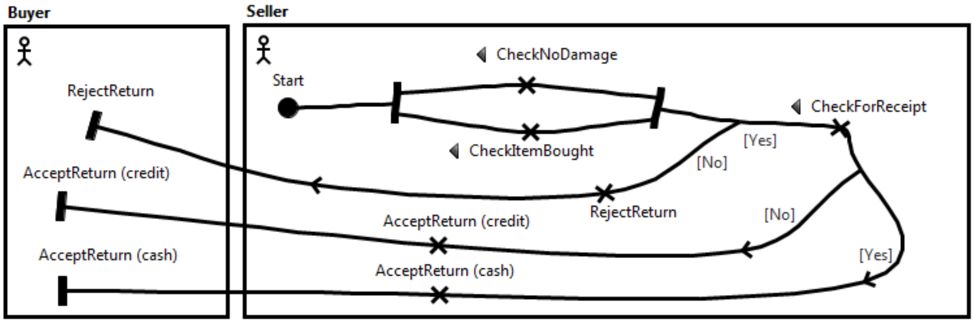

Figure 3: UCM Model to Check Acceptance Conditions

to reject or accept the return, depending on the result of the responsibilities. In the first case, the two responsibilities CheckForItemBought and CheckForReceipt are in an AND-fork, and if one of them is false, the return will be rejected. In the second case, if the responsibility CheckForDamage is true, the return is accepted.

3.3 Traceability in URN

The various GRL elements can be connected to UCMs in several of ways. Tasks in GRL can be linked to responsibility in UCM, or actors in GRL can be mapped to components in UCM. A UCM could also elaborate a collection of tasks (for instance, to explore ordering among these tasks) or be connected directly to goals or softgoals in the GRL model. For instance, the responsibilities CheckItemBought, CheckForReceipt, and CheckNoDamage in the UCM in Figure 3 are linked to the goals Bought Item, Have Receipt, and Not Damaged of the GRL in Figure 1, respectively. Also, the goals Return Product for Cash and Return Product for Credit are linked to the entire UCM in Figure 2. Such traceability relationships are important, especially during the evolutions of the system where they can be used for impact analysis.

A goal not covered by any scenario is a symptom of an incorrect or overspecified GRL models or of an incomplete UCM model. Similarly, a scenario that does not contribute to any goal is either not necessary, or the goal model needs to be enhanced.

4 Traceability of Arguments and Evidence in URN

As we have seen in the previous section, URN offers some traceability of UCM elements to GRL elements through URN links. This allows the user to verify the completeness and consistency of both the GRL diagrams and the UCM scenarios by checking for “gaps”. This also allows stakeholders to iteratively improve their model by working on both sides and constantly check whether there is still traceability between the two approaches.

URN provides some traceability of rationalization for GRL elements as well, namely using Belief elements. Beliefs can be seen as *arguments* for certain elements, which are the result of *discussions* between the stakeholders. Still, it seems that capturing a discussion using a single belief element is rather limited. This is also clearly seen from our running example. The discussion process between the stakeholders described in Section 2 shows that initially, the stakeholders agreed that they should not allow customers to return products when they do not have a receipt, but this argument was later defeated by another argument, namely that allowing customers to return products without a receipt for company credit leads to a bigger increase in customer service, and by the evidence that this does not occur very often.

The GRL language is not able to model such argument structures, because the beliefs elements in GRL are *flat*. There do not exist attack or support relations between belief elements, and beliefs cannot be supported by evidence either. Therefore, the actual discussion that occurred between stakeholders is lost, meaning that there is no traceability from these discussions to the resulting GRL elements. In this section, we propose a framework that extends URN with such argument structures. Our approach is based on the so-called *hybrid theory* used in reasoning about evidence in criminal cases [6].

4.1 The Hybrid Approach

Two main approaches exists in the literature on reasoning on the basis of evidence, namely the argument-based approach and the narrative-based approach. Recently, Bex et al. [5] have introduced a *hybrid approach*, in which arguments and narratives can be combined with each other in a single theory. This approach turned out to be a natrual way of modeling the process of proof, the iterative process of constructing, testing and justifying hypotheses in crime investigation and decision-making [7]. Moreover, it was used as the starting point for a sense-making tool, AVERS [8], in which the knowledge about a case can be mapped and visualized using both stories and arguments. We will now show how we can model our running example using the hybrid approach, and that it fits the URN specification very well. We will discuss the argument-based side of the hybrid approach, and next the narrative-based side. Finally, we discuss how to combine them.

In Figure 4, an example of an argument tree from the argument-based approach is given, which reflects part of the state of the discussion between the stakeholders in our running example of Section 2. The arrows between the premises and (intermediate) conclusions specify (defeasible) inferences. The coloured boxes (leaves of the tree) are *evidence* for the different arguments. This evidence can be an opinion of a group of stakeholders, or more formal evidence such as a document or a scientific article. For instance, evidence E1 can be a statement that all stakeholders have agreed upon such as the mission statement of the company. A defeasible inference is associated with an underlying generalization that acts as a *warrant* [16]. For example, the inference from evidence E1 to the argument "We want to increase profit" is justified by the generalization 'When something is stated in a mission statement of the company, the arguments holds', which can be rephrased as 'the mission statement mentions p, therefore (presumable) p'.

We can observe that this argument tree has links with GRL elements, which provides the means to trace back GRL elements to the argument tree. For instance, the GRL goal Return Product for money of Figure 1 can be traced back to the top argument, while the GRL softgoal Improve Customer Support can be traced back to the sub-argument "We want to increase customer support".

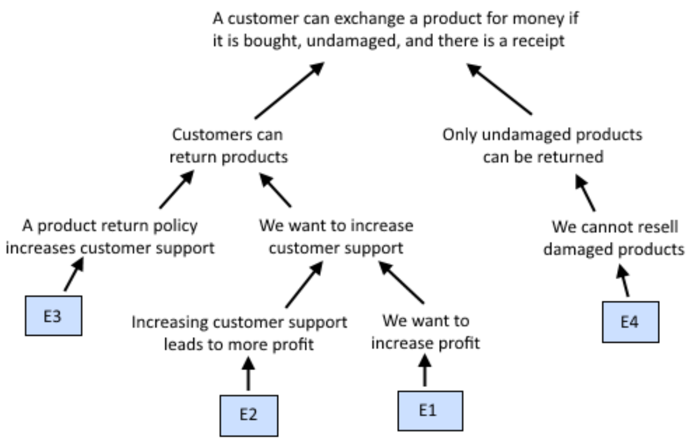

Figure 4: An argument based on evidence

Although Figure 4 only shows arguments that support each other, arguments can also be *attacked* by other arguments. They can be *rebutted* with an argument for the opposite conclusion and they can be *undercut* with an argument for why an inference is not allowed (this is mostly because some generalization does not apply in the given circumstances) [7]. In the example, an argument for "Customers cannot return products" rebuts the argument "Customers can return products" and an argument for "The mission statement is interpreted incorrectly" undercuts the inference from E1 to "We want to increase profit". These attacking arguments can again be attacked and thus the status of arguments (e.g. "justified", "overruled") can be determined dialectically.

Now that we have explained how we can integrate the argument-based side of the hybrid approach into URN, we will continue with the narrative-based side. The narrative-based approach used in reasoning about evidence in a criminal case is based on the observation that people tend to organize evidence by building stories about what might have happened [15]. Such stories are essentially network of events which are causally linked. This is very similar to the way that UCMs are built up (Figure 2 and 3). UCMs contain causally linked events that are called responsibilities. Therefore, the narrative-based side of the hybrid approach can directly be integrated into URN, simply by treating the UCMs as narratives.

The main idea of the hybrid approach is to link arguments in the argument-based approach with events in narrative-based approach. We can do the same thing for URN by treating the narratives as the UCMs. As we have explained above, argument trees have several points in which they can be linked with GRL elements. Moreover, GRL elements in turn have traceability links with URN elements as well (provided by URN, see Section 3.3). Thus, GRL elements provide the connection to trace UCM elements back to the arguments between the stakeholders and the underlying evidence.

5 Discussion

Having traceability from UCM elements to discussions between the stakeholders and the underlying evidence is useful for several purposes. First of all, URN elements can be firmly anchored or, in other terms, *evidentially supported*. As we mentioned in Section 3.3, the entire UCM of Figure 2 can be traced back to the goals Return Product for Money and Return Product for Company Credit, which in turn can be traced back to the argument "Customers can return products", which is based on several sub-arguments and evidence. Arguments can be attacked, which may break the "anchor's chain", causing the story to be no longer connected to the ground [7]. If, for example, the argument "Customers can return products" is defeated by another argument, then the corresponding UCM of Figure 2 is no longer supported by an argument. Aside from the anchoring stories in evidence, traceability from arguments between stakeholders to URNs also makes it possible to reason about the coherence of a URN model in a dialectical way.

Since our approach is very similar to the hybrid approach, we can use two types of criteria that are used in the hybrid approach to determine the quality of a narrative to determine the quality of an UCM. The first one is the extent to which a UCM conforms to the evidence (anchoring), and the second one is the coherence (consistency of causal relations). Bex [4] gives these and other criteria as a list of *critical questions*, typical sources of doubt when reasoning with the hybrid theory. The questions that concern the extend to which the story conforms to the evidence are as follows:

- *Evidential support* or *anchoring*: How much and which of the available evidence supports the story?
- *Evidential contradiction:* How much and which of the available evidence contradicts the story?
- *Evidential gaps:* How many and which events in the explanation are unsupported by evidential data?

The other three questions concern a story's coherence, the extent to which a story conforms to commonsense knowledge:

- *Plausibility:* How plausible are the events and causal relations in the story?
- *Completeness:* Does the story adhere to a plausible story scheme?
- *Consistency:* Are there elements of the story that contradict each other?

It is clear that we can almost directly translate these questions into our framework, with the exception of the question about completeness. This question mentions so-called *story schemes*, but we will not discuss them here.

6 Conclusion

In this paper, we discussed and analyzed URN modeling language for capturing rationale and design decision through a simple example. We identified the need for documenting evidences to help argumentation against or for rationales. Thus, we proposed to extend URN with a hybrid approach which helps in identifying evidence with both argument-based and narrative approaches.

7 Acknowledgments

This work has been partially funded by FNR PEARL programme and AFR - PDR grant #5810263, Luxembourg.

References

[1] Okhaide Akhigbe, Daniel Amyot, and Gregory Richards. A framework for a business intelligence-enabled adaptive enterprise architecture. In *33rd Int. Conf. on Conceptual Modeling (ER'014)*, LNCS, 2014.

[2] Daniel Amyot. Introduction to the user requirements notation: Learning by example. *Comput. Netw.*, 42(3):285–301, June 2003.

[3] Daniel Amyot, Sepideh Ghanavati, Jennifer Horkoff, Gunter Mussbacher, Liam Peyton, and Eric S. K. Yu. Evaluating goal models within the goal-oriented requirement language. *Int. J. Intell. Syst.*, 25:841–877, August 2010.

[4] Floris Bex. Analysing stories using schemes. *Legal Evidence and Proof: Statistics, Stories, Logic*, pages 93–116, 2009.

[5] Floris J. Bex, Henry Prakken, and Bart Verheij. Formalising argumentative story-based analysis of evidence. In *ICAIL*, pages 1–10. ACM, 2007.

[6] Floris J. Bex and SpringerLink (Online service). *Arguments, Stories and Criminal Evidence*. Law and Philosophy Library,.

[7] Floris Jurriaan Bex and Bart Verheij. Story schemes for argumentation about the facts of a crime. In *AAAI Fall Symposium: Computational Models of Narrative*, volume FS-10-04 of *AAAI Technical Report*. AAAI, 2010.

[8] S.W. van den Braak. *Sensemaking Software for Crime Analysis*. PhD thesis, Utrecht, The Netherlands, 2010.

[9] Okhaide Akhigbe et al. Creating quantitative goal models: Governmental experience. In *33rd Int. Conf. on Conceptual Modeling (ER'014)*, LNCS, 2014.

[10] D. Gross and E. Yu. Evolving system architecture to meet changing business goals: an agent and goal-oriented approach. In *Proc. 5th IEEE Int. Symposium on RE*, pages 316–317, 2001.

[11] ITU-T. Recommendation Z.151 (11/08): User Requirements Notation (URN) – Language Definition. `http://www.itu.int/rec/T-REC-Z.151/en`, 2008.

[12] Werner Kunz, Horst W. J. Rittel, We Messrs, H. Dehlinger, T. Mann, and J. J. Protzen. Issues as elements of information systems. Technical report, 1970.

[13] Lin Liu and Eric Yu. Designing web-based systems in social context: A goal and scenario based approach. In *Advanced Information Systems Engineering*, volume 2348 of *LNCS*, pages 37–51. Springer Berlin Heidelberg, 2002.

[14] Gunter Mussbacher and Daniel Amyot. Goal and scenario modeling, analysis, and transformation with jUCMNav. In *ICSE Companion*, pages 431–432, 2009.

[15] Nancy Pennington and Reid Hastie. Reasoning in explanation-based decision making. *Cognition*, 49(12):123 – 163, 1993.

[16] Stephen E. Toulmin. *The Uses of Argument*. Cambridge University Press, July 2003.

[17] M. Weiss and D. Amyot. Designing and evolving business models with URN. In *Proc. of Montreal Conference on eTechnologies (MCETECH05)*, pages 149–162, 2005.

[18] E. Yu, M. Strohmaier, and Xiaoxue Deng. Exploring intentional modeling and analysis for enterprise architecture. In *10th IEEE EDOCW '06. Int.*, pages 32–32, 2006.