iStar 2013 – Proceedings of the 6th International *i** Workshop

17th-18th June, 2013

Valencia, Spain

http://www.cin.ufpe.br/~istar13/

Jaelson Castro
Jennifer Horkoff
Neil Maiden
Eric Yu (eds.)

a CEUR Workshop Proceedings, ISSN 1613-0073

http://ceur-ws.org/Vol-978

Table of Contents

Scientific Papers

Session 5: Security & Risk

Session 6: Variability, Adaptation, Preferences & Qualities

Session 7: Know-how Mapping

Tool Fair Demos

Preface

The iStar workshop series is dedicated to the discussion of concepts, methods, techniques, tools, and applications associated with *i** and related frameworks and approaches. Following successful workshops in Trento, Italy (2001), London, England (2005), Recife, Brazil (2008), Hammamet, Tunisia (2010), and Trento, Italy (2011), the 6th International *i** Workshop is being held in Valencia, Spain. As with previous editions, the objective of the workshop is to provide a unique opportunity for exchanging ideas, comparing notes, and forging new collaborations.

This year, the workshop is co-located with the 25th International Conference on Advanced Information Systems Engineering (CAiSE'13), benefiting from the common interests shared by the workshop and the conference. As with past editions, we have tried to keep the format small and informal so as to maximize interaction. We are holding a discussion-oriented workshop, as outlined by the CAiSE'13 guidelines; as such the main criterion for paper acceptance in iStar'13 is relevance and potential for raising discussion. Presentation times for each regular paper, ranging from 10 to 20 minutes, have been split equally into presentation and discussion. The iStar Tool fair session included short presentations and 30 minutes for open demos and discussion. The program included a keynote given by Prof. Roel J. Wieringa entitled "The Role of Goals in Design Reasoning". A wrap-up session summarizing presented work and discussing areas of future investigation.

Concerning the review process, each of the 27 submitted papers went through a thorough review process with three reviews from a programme committee, providing useful feedback for authors. Revised versions of the 26 accepted papers are included in these proceedings. Proceedings include 20 regular and 6 tool papers. We thank authors and reviewers for their valuable contributions.

Last but not least, we want to deeply thank the organizers of the CAiSE conference for their great support.

We look forward to lively conversations and debates with old and new friends at the workshop, in the wonderful surroundings of Valencia, Spain!

Jaelson Castro, *Universidade Federal de Pernambuco, Brazil*
Jennifer Horkoff, *University of Trento, Italy*
Neil Maiden, *City University, London*
Eric Yu, *University of Toronto, Canada*

Programme Committee

Fernanda Alencar, *Universidade Federal de Pernambuco, Brazil*.
Daniel Amyot, *University of Ottawa, Canada.*
Luiz Marcio Cysneiros, *York University, Canada*.
Fabiano Dalpiaz, *University of Trento, Italy*.
Hugo Estrada, *CENIDET, Mexico*.
Paolo Giorgini, *University of Trento, Italy*.
Renata S.S. Guizzardi, *Universidade Federal do Espírito Santo, Brazil*.
Jennifer Horkoff, *University of Toronto, Canada*.
Dimitris Karagiannis, *Universitaet Wien, Austria*.
Alexei Lapouchnian, *University of Toronto, Canada*.
Julio Cesar Sampaio do Prado Leite, *Pontificia Univ. Catolica do Rio de Janeiro, Brazil*.
Sotirios Liaskos, *York University, Canada*.
Lin Liu, *Tsinghua University, China.*
James Lockerbie, *City University London. UK*.
Lidia López, *Universitat Politècnica de Catalunya, Spain*.
Oscar Pastor, *Universidad Politécnica de Valencia, Spain*.
Michael Petit, *University of Namur, Belgium*.
André Rifaut, *Centre de Recherche Public Henri Tudor*, Luexembourg.
Pete Sawyer, *Lancaster University, UK*.
Juan Carlos Trujillo, *Universidad de Alicante, Spain*.
Yves Wautelet, *Hogeschool-Universiteit Brussel, Belgium*.
Jelena Zdravkovic, *Stockholm University, Sweden*.

Steering Committee

Xavier Franch*, Universitat Politècnica de Catalunya, Spain*
John Mylopoulos*, University of Trento, Italy*

Submissions Management

Jennifer Horkoff, *University of Trento, Italy*.

Tools Fair Organizer

Jennifer Horkoff, *University of Trento, Italy*.

Webmaster

Tarcisio Couto Pereira, *Universidade Federal de Pernambuco, Brazil*.

The Role of Goals in Design Reasoning

Roel Wieringa

University of Twente, Department of Computer Science, Information Systems Group
P.O. Box 217, 7500 AE Enschede, The Netherlands
r.j.wieringa@utwente.nl

Designers reason from real or imagined stakeholder goals about a problem context, to desired properties of artifacts that should contribute to these goals in this context. The general pattern of reasoning is the same in software engineering, information systems and industrial product design: Given stakeholder goals G and assumptions about a context C, find artifact requirements R such that $C \times R \Rightarrow G$. Design reasoning is creative, as goals are usually not given ready-made to designers, the problem context is often partly unknown, assumptions about it are usually incomplete, and the artifact does not exist yet. Increased understanding of one of the three components (goals, context, artifact) changes the designer's understanding of the other two. This is not a stepwise refinement process but a non-monotonic process in which earlier beliefs may have to be retracted. The result, the contribution argument $C \times R \Rightarrow G$, is defeasible (it may turn out to be wrong).

After an analysis of design reasoning, I will zoom in on the role of goals in this kind of reasoning. I will define goals as stakeholder desires for which the stakeholder has committed resources (time and money) to achieve them. Stakeholders have different levels of goal awareness, ranging from unaware to actively pursuing the goal. Goals change, and in particular they can change by introduction of an artifact. Pursuing a goal entails having a problem theory that provides explanations, right or wrong, of the current state of the world, and predictions, right or wrong, about the future evolution of the world, and about the impact of different possible events on goal achievement.

I will end the talk by discussing implications for goal-oriented requirements specification languages such as i* at two levels. At one level, my analysis has implication about what aspects of goal-oriented design reasoning can be represented in a goal-oriented language. At another level, my analysis can be used to assess the role of i* as artifact used in a requirements context to contribute to goals of requirements engineers.

An Ontology-Based Methodology for Integrating *i** Variants

Karen Najera[1,2], Alicia Martinez[2], Anna Perini[3], and Hugo Estrada[1,2]

[1] Fund of Information and Documentation for the Industry, Mexico D.F, Mexico
{karen.najera, hugo.estrada}@infotec.com.mx
[2] National Center of Research and Technological Development, Cuernavaca, Mexico
amartinez@cenidet.edu.mx
[3] Bruno Kessler Foundation - IRST, Center for Information Technology, Trento, Italy
perini@fbk.eu

Abstract. Many variants of the *i** Framework have been developed since its definition. For instance, *i**-based modeling languages have been proposed for agent-oriented and service-oriented software development, for modeling risk, security requirements, and so on. The integration of models expressed in *i** variants poses interoperability problems. This issue has been faced at different levels, e.g. through unified metamodels, or with an interchange format to depict *i** models. In our research work, we propose a practical approach to tackle the interoperability problems, exploiting ontology-based techniques. In a previous work, we presented an ontological representation of the *i** metamodel and a tool to automatically transform an *i** model to an instance of this ontology. In this position paper, we describe a methodology to integrate *i** variants through the ontological representation of the *i** metamodel, and a mechanism to support the understanding of models expressed in those variants. As example of application, we present the integration of *i**, Tropos and Service-oriented *i** by using ontologies and a tool to automatically transform models expressed with those variants in terms of that ontology.

Keywords: *i**, organizational modeling, ontologies, model-driven engineering, model transformations.

1 Introduction

The *i** Framework [9] is a widely used organizational modeling technique. Since its definition, many research projects have used it in different application domains, hence many *i** variants have been proposed. For instance, Tropos [5] for agent-oriented development, Service-oriented *i** [3], and many others. Commonly, differences among variants consist of the addition of constructs or changes in the semantics of the existing ones in the original framework, as remarked also in [4], which refers to a study of 63 papers on *i** related methodologies, which were published between 2006-2010. Regardless of the degree of difference in variants, the integration and understanding of models expressed in *i** variants poses interoperability problems. This issue has been faced at different levels, e.g. through unified metamodels ([1, 6]), and with an interchange format to depict *i** models [2]. In this research work [7], we aim to show that the use of ontologies

can provide a practical approach towards tackling the *i** variants interoperability problem. As a preliminary stage, we have proposed the use of ontologies to integrate *i** variants, propitiating the understanding of the variants and their models by means of a common language established by the ontologies. We used the Web Ontology Language "OWL" to describe ontologies, since it is a standard of the Semantic Web which facilitates greater machine interpretability than XML or RDF, and provides a formal semantics. With our proposal, we bring the advantages of ontologies to the organizational modeling domain, such as querying, reasoning and organizational data specified in a Semantic Web format. Our initial results have been presented in [8], namely: a) The development of an ontology-based metamodel for *i** called OntoiStar and b) a tool-supported process to generate ontologies from models expressed in *i**. In this paper, we present: a) a methodology to integrate *i** variants by using ontologies on the basis of OntoiStar, b) an example of application integrating the variants *i**, Tropos and Service-oriented *i**, and c) the extension of our tool-supported process to support the understanding of models expressed with the integrated *i** variants.

2 Objectives

The main objective of our research work [7] is *The integration of i* variants through the use of an ontology and the automatic representation of models expressed with those variants in terms of the ontology with* i* *variants integrated. Thereby supporting the understanding of variants and their models.* For the fulfillment of this objective four specific objectives have been identified:

1. The development of an ontology called OntoiStar for representing the core constructs of the *i** variants.
2. The development of a methodology for integrating the constructs of several *i** variants through the use of ontologies the basis of the ontology OntoiStar.
3. The application of the methodology to different i* variants, generating an ontology with *i** variants integrated.
4. The automatic transformation of a model represented with one of the integrated *i** variants into an ontology derived from the concepts of the ontology with the *i** variants integrated.

The first specific objective has been addressed in [8]. In this paper we focus on the last three specific objectives.

3 Scientific contributions

Our scientific contributions are related to the accomplishment of the specific objectives 2, 3 and 4 of our research work. Therefore, in this section we present our proposed methodology, which describes how to use ontologies to achieve the integration of *i** variants. Moreover, we present the application of the methodology to *i**, Tropos and Service-oriented *i**.

3.1 Integration methodology

Our proposed integration methodology provides the guidelines to integrate constructs of several *i** variants into an ontology which extends OntoiStar [7, 8].

The ontology with i^* variant integrated has been named OntoiStar+ in a general way, to indicate this ontology considers the constructs of two or more i^* variants, no matter which or how many they are. The integration methodology to obtain OntoiStar+ is based on Model-Driven Engineering (MDE), since it is generated on the basis of OntoiStar which was developed through MDE at the level of metamodels. The methodology consists of two phases: 1) the development of an ontology for each i^* variant desired to integrate and 2) the integration of the ontologies of the i^* variants generating the ontology OntoiStar+.

Phase 1) Development of an ontology for a specific *i variant**
In this phase the ontology for a specific i^* variant is generated. The resultant ontology is based on OntoiStar. Therefore, additional constructs of a specific i^* variant are added into OntoiStar. This phase may be performed several times when more i^* variants need to be integrated. It consist of four steps:

I. Identify. The first step corresponds with the identification of the additional constructs of the i^* variant which are not part of the ontology OntoiStar.

II. Categorize. The second step corresponds to the categorization of the additional constructs identified in the first step. Four categories allied with the elements of metamodels have been defined. A construct is categorized as:
Concept, when it corresponds to a representation of something in the real world.
Relationship, when it corresponds to a relationship of two or more concepts.
Attribute, when it is used to define a property or characteristic of a concept. It may also refer to or set the specific value for a given instance of such.
Attribute value, when it corresponds to those values that belong to an additional construct which has been categorized as attribute.

III. Transform. The third step corresponds to the transformation of the additional constructs into ontological constructs, i.e. classes, properties and axioms in OWL. Two sets of transformation rules have been proposed according with the i^* variant specification. One for those constructs identified from a metamodel described in the UML language, and other for constructs identified from a textual description (Table 1).

IV. Classify. In [7, 8] we presented the OntoiStar taxonomy which was defined according with the core i^* concepts specified in [2]. This step corresponds with the classification, within OntoiStar taxonomy, of the new OWL classes resulting from the third step. Therefore, a new OWL class is subclass of the class:
Actor, if the OWL class describes a new type of actor.
Actor Relationship, if the OWL class describes a new type of actor relationship.
Dependency, if the OWL class describes a new dependency relationship. The dependency basic structure has been already defined in OntoiStar.
Boundary, if the OWL class describes a new type of boundary.
Intentional Element, if the OWL class describes a new type of intentional element.
Intentional Element Relationship, if the OWL class describes a new type of intentional element relationship.

Table 1. Transformation rules

From metamodel	From textual description
Each concept, concept relationship and enumeration class is represented as a class in OWL.	Each concept, concept relationship and attribute is represented as a class in OWL.
Each association is represented as an object property in OWL.	If a class in OWL is created due a concept relationship, two object properties are created to complete the relationship.
Each class property is represented as axioms in OWL.	-
Each enumeration element is represented as a class instance of the owner enumeration class in OWL.	If a class in OWL is created due an attribute, each attribute value is represented as a class instance of the corresponding attribute class in OWL.
Attributes. Each enumeration type is represented as an object property in OWL. Each primitive data type is represented as a data property in OWL.	If a class in OWL is created due an attribute, an object property is created to complete the representation of the attribute.

Phase 2) Integration of the ontologies of the *i variants**
In this phase, the ontologies of the i^* variants (generated in phase 1) are integrated by means of an iterative merging process, obtaining as a result, the ontology OntoiStar+. The merging process consist of applying a merging function to the i^* variant ontologies, two at a time, till obtain the ontology with all the desired i^* variants. It is applied first to two i^* variant ontologies, then, the resultant ontology is merged with another i^* variant ontology, and so on. See for instance Fig. 1. The merging function consist of bringing together all the constructs of two ontologies, taking into account that duplicated constructs are only considered one time in the final merged ontology. The merging function has been implemented in the tool described in 3.3. With this approach a user can select the ontologies to merge according to the i^* variants the user works with.

3.2 Application of the integration methodology

We have applied the integration methodology to i^*, Tropos and Service-oriented i^*. The integration results are presented in Table 2 (attributes were omitted due to space). The first two columns describe constructs already included into OntoiStar. The next three columns contain the additional constructs of each variant (N.A. indicates there is no additional constructs). First, we performed Phase 1, therefore, additional constructs of each variant were identified, categorized, transformed and classified in order to obtain the ontology for each variant. For instance, in Service-oriented i^*, the construct "Service" was identified, then, it was categorized as concept and transformed as an OWL class. Finally, it was classified as an Intentional Element, therefore, placed as a sub class of the Intentional Element class in the OntoiStar taxonomy. Then, Phase 2 was carried out by merging the ontology for i^*, the ontology for Tropos and the ontology for Service-oriented i^*, obtaining as a result the ontology OntoiStar+ with the three variants integrated. The integration process is represented in Fig. 1. The

Table 2. Additional constructs of each *i** variant.

OntoiStar		***i****	**Tropos**	**S-O *i****	**OntoiStar+**
***i** concept**	**Types**	**Types**	**Types**	**Types**	**Types**
Actor	Agent, Role, Position	N.A.	N.A.	N.A.	Agent, Role, Position
Actor Relationships	Is_part_of, Is_a, Plays, Covers, Occupies	Instance_of	N.A.	Subordination	Is_part_of, Is_a, Plays, Covers, Occupies, Instance_of, Subordination
Dependency	Goal, Softgoal, Task, Resource	N.A.	Plan	Service, Process	Goal, Softgoal, Task, Resource, Plan, Service, Process
Boundary	-	N.A.	N.A.	N.A.	-
Intentional element	Goal, Softgoal, Task, Resource	N.A.	Plan	Service, Process	Goal, Softgoal, Task, Resource, Plan, Service, Process
Intentional element relationship	Contribution, Decomposition, MeansEnd	N.A.	N.A.	Service, Service-goal, Process	Contribution, Decomposition, MeansEnd, Service, Service-goal, Process

last column of Table 2 contains the constructs included in the resultant ontology OntoiStar+.

3.3 Automatic transformation from *i** based models to ontologies

In this section, we present TAGOOn (Tool for the Automatic Generation of Organizational Ontologies), an extension of the one in [8]. TAGOOn provides the basis to support the automatic transformation from models expressed with different *i** variants in ontologies. The current version supports models expressed with *i**, Tropos and Service-oriented *i**. The transformation process starts with the representation of models using iStarML [2]. Some additional constructs of the supported variants are not defined in its grammar. Therefore, we have used the open options of the iStarML specification to represent them. For instance, using the attribute 'type', concepts as plan, service and process could be defined in the tag $< ielement >$; and relationships as service relationship and service dependency could be defined using in the tag $< ielementLink >$.

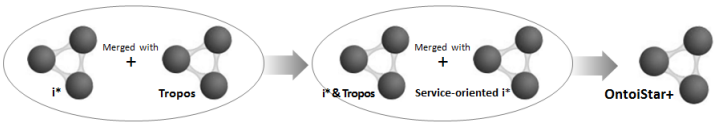


Fig. 1. *i**, Tropos and Service-oriented *i** integration process

In order to perform the automatic transformation, TAGOOn parses the iStarML file and, according to pre-defined mapping rules, it instantiates the corresponding classes and properties in the ontology OntoiStar+. The output of the tool corresponds to the ontology OntoiStar+ with instances that represent the knowledge depicted in the organizational model. It shapes an organizational knowledge base in which is possible to apply services offered by ontologies such as querying and reasoning. Furthermore, it can be edited with an ontology editor, or it can be the input of development or reasoning platforms supported by ontologies.

4 Conclusion and Future Work

In this paper, we presented a further step in the achievement of interoperability of *i** variants, which extends a previous work were an ontological representation of the *i** metamodel was proposed [8]. Specifically, we presented a methodology to integrate several *i** variants into an ontology and our supporting tool TAGOOn. With our proposal, we bring advantages of ontologies such as querying and reasoning, to the organizational modeling domain. Moreover, as the organizational knowledge is represented in OWL, it could be available to be exploited and consumed in the Semantic Web by paradigms such as Linked Data.

The main steps we intend to address in our future work can be summarized as follow: a) To take into account the semantic of the *i** variants during the creation of ontologies; b) To propose inference rules that enable the redefinition and adjustment of *i** based models according with the semantic and differences of *i** variants; c) To extend TAGOOn to carry out the automatic transformation of *i** based models from one *i** variant to other *i** variant.

References

1. C. Cares, X. Franch, E. Mayol, and C. Quer. A Reference Model for i*. In *Social Modeling for Requirements Engineering*, pages 573–606. MIT Press, 2010.
2. C. Cares, X. Franch, A. Perini, and A. Susi. Towards interoperability of i* models using istarml. *Computer Standards & Interfaces*, 33(1):69–79, 2011.
3. H. Estrada. *A service-oriented approach for the i* framework*. PhD thesis, Valencia University of Technology, Valencia University of Technology, Valencia, Spain, 2008.
4. X. Franch. The i* framework: The way ahead. In *Sixth International Conference on Research Challenges in Information Science RCIS'12*, pages 1–3, 2012.
5. P. Giorgini, J. Mylopoulos, A. Perini, and A. Susi. The Tropos Methodology and Software Development Environment. In *Social Modeling for Requirements Engineering*, pages 405–423. MIT Press, 2010.
6. M. Lucena, E. Santos, C. T. L. L. Silva, F. M. R. Alencar, M. J. Silva, and J. Castro. Towards a unified metamodel for i*. In *Research Challenges in Information Science*, pages 237–246, 2008.
7. K. Najera. An ontology-based approach for integrating i* variants. Master's thesis, National Center of Research and Technological Development, Cuernavaca, Morelos, Mexico, 2011. `www.tagoon.semanticbuilder.com/NajeraThesis.pdf`.
8. K. Najera, A. Perini, A. Martínez, and H. Estrada. Supporting *i** model integration through an ontology-based approach. In *iStar*, pages 43–48, 2011.
9. E. S.-K. Yu. *Modelling strategic relationships for process reengineering*. PhD thesis, University of Toronto, University of Toronto, Toronto, Ont., Canada, 1996.

Extension and integration of i* models with ontologies

Blanca Vazquez[1,2], Hugo Estrada[1], Alicia Martinez[2], Mirko Morandini[3], and Anna Perini[3]

[1] Fund Information and Documentation for the industry - INFOTEC, Mexico
{blanca.vazquez, hugo.estrada}@infotec.com.mx
[2] National Center of Research and Technological Development - CENIDET, Mexico
{blancavazquez11c, amartinez}@cenidet.edu.mx
[3] Bruno Kessler Foundation - IRST, Center for Information Technology - FBK, Italy
{morandini,perini}@fbk.eu

Abstract. Currently, i* is one of the most well founded organizational modelling techniques. Its main feature is the expressibility to represent intentional social relations among stakeholders. In i* models, each modeling component is described explicitly through text labels. However, the process of labeling model elements is usually an activity which is not rigorous and not well documented for designers. Performing the labeling with freedom and subjectivity often results in unclear labels that are not helpful for interoperability and the understanding of the model semantics. In this paper, we deal with this problems by extending i* models with ontologies. Taking advantage of an ontological definition of concepts and well-defined relationships, we improve the unambiguous interpretation of labels and thus interoperability, reuse and machine-readability of a model. A guided process and tool support for the integration of i* models with an ontology are described in this paper.

Keywords: conceptual modeling, iStar, semantic annotation, ontology.

1 Introduction

The i* framework is a goal-oriented and agent-oriented modeling framework. It provides the needed infrastructure to model concepts such as actors, roles and agents, and to reason about them. Nowadays, many research projects exist that use the i* in different applications domain on early requirements engineering, business process design and system requirements [2]. The i* framework defines two key models at different level of abstraction: the Strategic Dependency and the Strategic Rationale model. A set of modeling primitives defines the model components and the relationships among them, where each business element is labeled according to its description. This labeling is usually the only reference in the model to indicate, to the analyst, the meaning of a specific model element.

However, the absence of guidelines or good-practices to label business elements usually leads to the subjectivity, resulting in ambiguous labels that make the models difficult to understand for both the analysts and the target audience.

Furthermore, the amount of information that can be encoded in a human readable label is necessarily limited. Thus, the interpretation of an organizational model can become inevitably complex. Moreover, the machine-readability of a model remains quite limited.

In the remainder of this paper we present an approach to extend i* models with semantic annotations taken of general or domain ontologies. This allows us the standardization of concepts, clarifying the labels that describe an element and also it permits to improve the analysis of existing models. Section 3.1 describes a set of semantic suggestions to guide the process of model annotation based on domain and general ontologies. Section 3.2 describes a tool-supported approach for combining the i* model (in its ontological representation [6]) with the ontology used for the annotation.

2 Objectives of the research

As first objective of this work, we propose the extension of i* models with concepts taken from an ontology in order to address the above-mentioned ambiguity issues that emerge from the labeling of organizational models.The enrichment of models with well-defined concepts allows us to clarify the labels that describe an element to improve the analysis of existing models. Starting from this, as a second objective, we try to explore the possibilities given by a (tool-supported) ontological representation of the annotated i* model joined with an ontology.

To address our first objective, we extend i* models with annotations from a well-defined ontology. The extension process consists of three steps *(i)* we describe a set of suggestions applicable to ontologies, these suggestions are the key to annotate the i* models; *(ii)* we propose an extension of the iStarML [3] model interchange format to represent the annotated model in a validated language, and *(iii)* we provide tool support for annotation, by extending the model export plug-in for *iStarML* of the jUCMNav i* modeling tool.

To address our second objective, we translate the i* model into an ontology [6],and we join this ontology with a general or domain ontology used in the annotation process. In this ontology, the model annotations provide a formal link between concepts and the instances of the model and thus collocate the model in the domain in an unambiguous way. Our approach can also be applied to models described in Tropos [1] and Service-oriented i* [4].

3 Scientific contributions

The core of our contribution is the extension of i* models with semantic annotations taken of ontologies. The top of Fig. 1 presents the proposed approach and the bottom side presents an example of this approach.

3.1 Model Annotation Process.

This process is composed by three main steps.

Step 1. Semantic annotation suggestions. To annotate model elements with concepts from an ontology, we first define guidelines, in the form of a set of semantic annotation suggestions. An ontology represents knowledge as concepts and relationships between them. Specifically, general ontologies such as DOLCE

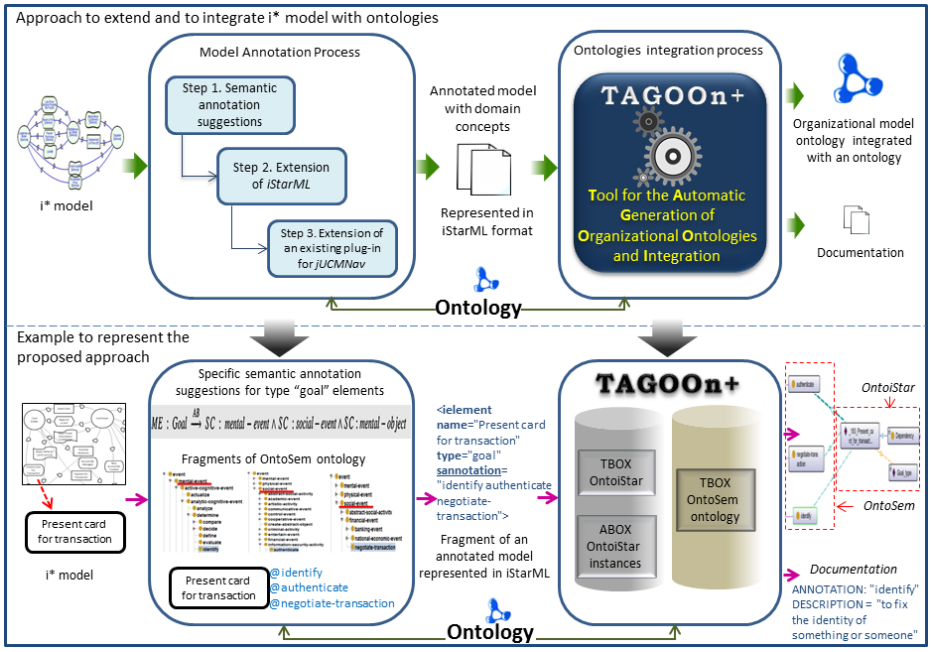


Fig. 1. Proposed approach to extend and integrate i* models and an example.

and UFO describes general concepts like space, time, matter, event, etc. [5], while the domain ontologies describe a vocabulary related to a limited domain (e.g., healthcare). In this work we use the *OntoSem* ontology [7]. OntoSem is a general ontology developed with a focus on practical application, basing on the root concepts of *object*, *event* and *property*.

To develop the suggestions, a semantic analysis of the primitives in i* and its variants was carried out. We analysed and compared the definitions among primitives of the same type (e.g., *goal*), in order to identify the differences and similarities among them, obtaining a single definition for each primitive. Next, analysing the structure of general and domain ontologies we tried to find matches between the obtained definition of the primitives and the concepts and relationships of the ontologies, with the goal to formally establish the relation of each obtained definition with ontology concepts. We provide a set of general semantic annotation suggestions (Table 1) and a set of specific semantic annotation suggestions (Table 2). The general suggestions are applicable to any general ontologies, while the specific suggestions are applicable to the OntoSem ontology and its extensions.

We illustrate the annotation process with two shorts examples. Let's assume that a goal element labeled *"Get credit"* is annotated using a domain ontology on financial operations. Using the general suggestions for elements of type *"goal"*(Table 1), the goal *"Get credit"* can be annotated with *"Financing"* and *"Credit request"*. Now, let's assume that a goal element is labeled *"Present card for transaction"* (bottom side of Fig. 1) using OntoSem and the

suggestions for elements of type *"goal"* (Table 2), where *"ME"* means *Model Element*, "$\xrightarrow{AB}$" means *can be annotated* and *"SC"* means *SuperConcept*, this element could be annotated with *"Negotiate transaction"*, *"Authenticate"* and *"Identify"*. The idea is to first follow the semantic suggestions, then to go indeep in the selected ontology and to find out the most appropriate concept for each model element, in a manual process. This approach could be used with different ontologies independently of the model domain thanks to the semantic annotation suggestions.

Table 1. General semantic annotation suggestions applicable to domain ontologies [8]

Primitive	Suggestion
Actor	An actor (including the actor types) should be mapped into domain concepts that describe an organization, agent, tangible entity, or intangible entity.
Goal	A goal should be mapped into domain concepts that describe a clear and precise condition, interest or desire.
Softgoal	A softgoal should be mapped into domain concepts that describe an interest or desires not clear-cut satisfaction criteria.
Task	A task should be mapped into domain concepts that describe a concrete action or activity.
Resource	A resource should be mapped into domain concepts that represent a physical object or informational entity.

Step 2. Extension of iStarML. To provide a format for interoperability between *iStar* modelling tools and dialects, we extend the iStarML model interchange format, adding an XML attribute called *sannotation*. This attribute stores the semantic annotations for each model element, with the syntax *sannotation* = *"concept*$_1$ *concept*$_2$*...concept*$_n$*"*.

Step 3. Extension of an existing plug-in for jUCMNAv. To use semantic annotation in practice, we use the popular iStar modeller jUCMNav, adding the annotations with the symbol "@". We extended an existing plug-in for jUCMNAv to generate the iStarML files containing the annotations.

3.2 Ontologies integration process

We provide a tool-supported process for joining annotated i* models with general or domain ontologies. First, an organizational ontology is created from an i* model by following the approach presented by Najera et al. [6]. Starting from models iStarML format, models are transformed to an OWL-based representation in the metaontology *OntoiStar*, defined for representing the i* metamodel. We propose to combine the *OntoiStar*-based ontology of an i* model with a domain or general ontology. The two ontologies will have two joint points: on one side, the foundational concepts used in both ontologies, and, on the other side, the strong connection created by the semantic annotation of the model elements with concepts provided by the selected ontology.

We enhanced the TAGOOn tool (*Tool for the Automatic Generation of Organizational Ontologies*) provided by [6] to achieve our objective. The resulting tool, called **TAGOOn+**, takes as inputs the semantically annotated iStarML model and the selected ontology in OWL format (see bottom side of Fig. 1).

Taking an i* model (M_{iStar}), TAGOOn+ creates its ontological representation (O_M) by using the functionalities provided by the original TAGOOn tool,

Table 2. Examples of specific semantic annotation suggestions for *model elements* (ME) using OntoSem *super-concepts* (SC), for OntoSem and its extensions [8]

Merging axioms	Intuitive meaning
$ME : Actor \xrightarrow{AB} SC : object$	A model element of type actor can be annotated only with (can represent only) the super-concept *object*.
$ME : Goal \xrightarrow{AB} SC : mental\text{–}event \wedge SC : social\text{–}event \wedge SC : mental\text{–}object$	A model element of type goal can be annotated only with (can represent only) the super-concepts mental-event, social-event and mental-object.
$ME : SoftGoal \xrightarrow{AB} SC : abstract\text{–}object$	A model element of type softgoal can be annotated only with (can represent only) the super-concept abstract-object.
$ME : Task \xrightarrow{AB} SC : active\text{–}cognitive\text{–}event \wedge SC : social\text{–}event \wedge SC : physical\text{–}event$	A model element of type task can be annotated only with (can represent only) the super-concepts active-cognitive-event, social-event and physical-event.
$ME : Resource \xrightarrow{AB} SC : physical\text{–}object \wedge SC : mental\text{–}object$	A model element of type resource can be annotated only with (can represent only) the super-concepts physical-object and mental-object.

instantiating each model element to an individual in O_M. Next, it analyzes the selected ontology (O_D) to obtain the hierarchy and description of each concept. O_M and O_D are joined, determining if an individual element was annotated with concepts (classes of the selected ontology). If an element is related with one or more concepts, a relation of type *is-a* is created between the concept in O_D and the instantiation of the model element in O_M. The obtained combined ontology (O_I) thus integrates the domain knowledge of O_D and the organizational and intentional perspective provided by M_{iStar}. For instance, a model element M_E is annotated with the concept C_1 from an ontology O_1. After the integration, the M_E is an individual element I_E that presents a relation of type *is-a* with C_1. The resulting ontology in OWL format can be visualized and edited with the popular Protègè ontology editor. In Figure 1 (bottom right side) fragments of OntoSem and i* models are visualized, both integrated into a single ontology.

Depending on the properties of the domain or general ontology used, Protègè reasoning engines offer consistency checking, inference and classification. Moreover, TAGOOn+ can create a text document that describes each element of the model with its semantic annotation and its description taken from the selected ontology. The documentation can be useful for achieving a better understanding of the i* model, for sharing and reuse.

Our approach has been used to annotate models that described a farm systems, a generic card-based payment system, and the processes to register students at a postgraduate institution [8]. In all these case, the annotations were validated by domain experts. The semantic annotations were helpful to discover hidden relationships, to collaborate with experts, to improve the understanding of the model and to eliminate ambiguity in labeling and thus to facilitate knowledge sharing.

4 Conclusions

In this paper we presented an overview of our approach to extend i* models with concepts taken from ontologies. A set of semantic suggestions guide the

annotation of i* models were presented. A model with semantic annotations from an ontology is clear for humans and accessible to machines. Moreover, it can help improving the labeling quality in particular for models that evolve for a long time or that need to be read or edited by various analysts and stakeholders with a different background. Joining the model (in its ontological representation) with a generic or domain ontology, depending on the properties and axioms defined in the ontology adopted, the obtained joint ontology could be checked for consistency and completeness. We provide tool support for the complete process and tested it with available *iStar* models and ontologies obtained from web repositories. The benefits of extending an i* model with annotations would be, first, to facilitate the analysis and understanding of a model proving a clear model supported for domain concepts and second, as our approach is based on iStarML, we permit the interoperability among i* variants through domain concepts.

5 Ongoing and future work

At the present time, we are focusing on extending i* by describing its elements with generic concepts. As a future work we attempt to use natural language processing techniques for the annotation of each model element. In this way, semi-automatic suggestions will be provided to annotate the model. Moreover, we consider that a concept that integrates different model elements could represent new business services to the organization. These new functionalities can be useful to delineate new business services, and to improve the understandability and expressiveness of a model, thus giving a necessary condition for model reuse.

Finally, an empirical study and the modelling of real-world systems would be needed to give practical and statistical evidence to the efficiency of the approach.

References

1. P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos. Tropos: An Agent-Oriented Software Development Methodology. *IJAAMAS*, 8(3):203–236, 2004.
2. C. Cares and X. Franch. A metamodelling approach for *i** model translations. In *CAiSE*, pages 337–351, 2011.
3. C. Cares, X. Franch, A. Perini, and A. Susi. iStarML: An XML-based Model Interchange Format for i*. In *iStar'08*, volume 322 of *CEUR Workshop Proceedings*, pages 13–16. CEUR-WS.org, 2008.
4. H. Estrada. *A service-oriented approach for the i* framework*. PhD thesis, University of Technology, Valencia, Spain, 2008.
5. N. Guarino. *Formal Ontology in Information Systems*. IOS Press, Amsterdam, The Netherlands, The Netherlands, 1st edition, 1998.
6. K. Najera. An ontology-based approach for integrating i* variants. Master's thesis, National Center of Research and Technological Development, Cuernavaca, Morelos, Mexico, 2011. `www.tagoon.semanticbuilder.com/NajeraThesis.pdf`.
7. S. Nirenburg and V. Raskin. *Ontological semantics*. MIT Press, 2004.
8. B. Vazquez. Enriching organizational models through semantic annotation. Master's thesis, National Center of Research and Technological Development, Cuernavaca, Morelos, Mexico, 2012. `www.tagoon.semanticbuilder.com/VazquezThesis.pdf`.

Using a Foundational Ontology to Investigate the Semantics Behind the Concepts of the *i** Language

Renata Guizzardi[1], Xavier Franch[2], Giancarlo Guizzardi[1] and Roel Wieringa[3]

[1]Ontology & Conceptual Modeling Research Group, Vitória, Brazil
{rguizzardi,gguizzardi}@inf.ufes.br
[2]Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
franch@essi.upc.edu
[3]University of Twente, Enschede, The Netherlands
roelw@cs.utwente.nl

Abstract. In the past few years, the community that develops *i** has become aware of the problem of having so many variants, since it makes it difficult for newcomers to learn how to use the language and even to experts to efficiently exchange knowledge and disseminate their proposals. Moreover, this problem also delays the transfer of the *i** framework to industrial settings. Our work is one of the current attempts to promote interoperability among the existing variants, and it does that by investigating the semantics behind the *i** core concepts. For that, we apply a foundational ontology named UFO, which is used as a semantically coherent reference model to which the language should be isomorphic. In this paper, we report on the steps we have pursued, what we have accomplished so far, also setting the context for the work ahead.

Keywords: iStar, language interoperability, semantics, foundational ontology, UFO

1 Introduction

Nowadays, the community that develops *i** is relatively big and these developers, who are geographically dispersed, tend to ascribe different (and sometimes conflicting) meanings to its constructs. It is argued that this flexibility is part of the framework's own nature, and in fact may be considered one of its key success features. But on the other hand, it is our belief that this represents a clear risk in terms of promoting the framework, creating serious issues, such as: a) hampering the efficient communication of knowledge among experts of the community [1]; b) increasing the learning curve of newcomers; and c) inhibiting the adoption of the framework by practitioners. We refer to [1][2][3] for a more detailed state of the art on the different uses and ascribed semantics of the *i** language constructs.

In the past few years, the community has become aware of this problem and several attempts have been made for facilitating the access and uniform use of the *i** language. One of these initiatives is the creation of a common repository and collaborative environment, namely the *i** wiki (http://istar.rwth-aachen.de/tiki-view_articles.php). In particular, there is a session in the wiki called *Guidelines*,

aimed at collecting and making explicit the different approaches to the language. Works on metamodeling have also tried to make it clear the meaning ascribed to the distinct constructs [3][4]. Although we recognize there are significant outcomes of these works (e.g. pointing out the applied concepts in particular variations; showing the author's view on how concepts relate), these attempts did not quite succeed in providing interoperability, simply because metamodels are powerful structures to define a language's syntax while being very limited in terms of clarifying its semantics. Cares [5] has proposed an interoperability method that considers a supermetamodel [6], which facilitates the translation from an *i** variant to another, and an XML-based mark-up language, named iStarML [7], which triggers existing tools to interoperate as much as their underlying metamodel allows. This approach has advanced the state of the art, by providing a standard interoperability format that facilitates model translation, but we are afraid that iStarML only makes syntactic checks, leaving the semantic interoperability issues still untouched.

Going beyond syntactic issues, since 2006, we are involved in an attempt to define a common ontology for the core concepts of the *i** language. We believe this may assist in clarifying the semantics of the language's concepts, thus generating a number of modeling guidelines targeted at enhancing the language's usability. We do this by applying the UFO foundational ontology as a reference model and our approach prescribes that the *i** metamodel reflects such model [8]. For that, at times, we propose some of the *i** constructs may be left aside, as they have the same semantics, thus being excessive in the language [9]. To diminish ambiguity, other times, we perform some language extensions, avoiding a single concept to be overloaded with two or more different semantics. However we attempt to do that with extra care, as it is undesirable to end up with a language with too many constructs as it would require even more effort to be learned and used.

This paper reports on what we have accomplished so far and also presents what lies ahead of us. For that, the remainder of the paper is structured in three sections besides this introduction: section 2 presents the objectives and methodology of our research, also pointing at which stage we currently are; section 3 describes the main contributions and challenges involved in this initiative; section 4 discusses some conclusions and presents our future research agenda.

2 Objectives and Methodology

The general objective of our long-term joint research is to provide an ontological foundation to the *i** language core. In the current research stage, we are clarifying the semantics of *i** intentional element links: means-end, decomposition and contribution. In this context, it is necessary not only to determine accurately the meaning of these constructs, but also to provide methodological advice on their use, since in some cases differences may be a bit subtle. To sum up, the general objective of understanding the **meaning of *i* intentional element links*** can be refined into a series of more concrete research questions: 1) For every type of intentional element link: Which are the logical conditions and implications of a link established from an intentional element to another? 2) In particular, which *i** intentional elements may appear as root and

target of such an intentional element link? 3) Which ontological properties must these elements fulfil? 4) Which are the methodological guidelines that drive to the application of a particular type of intentional element link? 5) What is the modeler's agreement on the aspects above? The rest of the paper reports the current results obtained so far and outlines the most immediate future work in relation to these research questions.

3 Scientific Contributions and Challenges

Our work has started with the semantics analysis of the core *i** intentional elements, such as ***actor, goal, task*** and ***resource***[1]. The results of this analysis are reported in [9]. This initial analysis has also taken into consideration the concepts of ***agent, role*** and ***position***, along with the ***dependence*** relation. More recently, our attempts shifted to the remaining relations of the language. In [8][10], we propose some modeling guidelines for the means-end link and OR-decomposition, based on UFO's semantic interpretation. According to UFO, a `goal` is the `propositional content` of an `agent's intention`. Thus, ontologically, a ***goal*** is in itself a *proposition* and ***decomposition relations*** reflect logical relations between propositions. Table 1 formally describe the And and OR decomposition of goal G in four subgoals G1-G4

Table 1. Formal Description of And and OR-decomposition according to UFO

AND-decomposition	$G \leftrightarrow G1 \wedge G2 \wedge G3 \wedge G4$
OR-decomposition	$G \leftrightarrow G1 \vee G2 \vee G3 \vee G4$

The *i** literature shows that modelers sometimes use ***OR-decomposition*** and ***means-end*** to express the same phenomenon. This is also reflected in some dialects. For instance, GRL does not allow ***OR-decomposition***, thus only means-end links are applied [11]. On the other hand, it is also very common to find two diagrams of the same modeler in which the same relation express distinct semantics [8].

In our work, we see the ***means-end link*** and ***OR-decomposition*** as two different relations. For example, a goal is as mentioned, a proposition, thus it is not possible to decompose a goal into tasks or resources. A goal may be only decomposed into subgoals. The ***means-end link***, on the other hand, is generally applied between ***tasks*** and ***goals***, for example. But how can we formally define the ***means-end link***? To understand that, one must first review the ontological meaning of intention. `Intentions` are mental states of `Agents` which refer to (are about) certain `Situations` in reality (i.e. states of affairs the `agent` aims at achieving). Now, we may define the notion of `deliberately achieving` a goal as follows:

$$task(a) \wedge goal(G) \wedge deliberately\text{-}achieves(a, G) \leftrightarrow$$
$$achieves(a, G) \wedge (\exists i{:}\ intention(i) \wedge is\text{-}reason\text{-}for(i, a) \wedge implies(propositional\text{-}content(i), G))$$

[1] From now on, we use different font to highlight *i** and UFO elements. For i* elements, we use bold and italic ***arial*** while for UFO elements, we apply `courier new`

In other words, a task *(*`action` in UFO*) a* `deliberately achieves` a `goal` *G* iff this task achieves *G* (i.e., causes the world to be in a state which makes *G* true) but also this task must be motivated by an `intention` (intention *i* is the reason for task *a*) whose `propositional content` implies *G*. Informally, we can state that *a* is performed with the intention of achieving *G*. Now, we are able to define the ***means-end link*** as:

$$task(a) \wedge goal(G) \wedge ME(a, G) \rightarrow deliberately\text{-}achieves(a, G)$$

In other words, a means-end link between a task *a* and a goal *g* holds if the execution of such task leads to goal achievement and is deliberately performed by the agent in whose perspective the relation is defined. Please note that this definition makes a clear distinction between the ***means-end link*** and the ***OR-decomposition*** relation, previously defined.

Another common case of *construct overload* in *i** is given by the indistinct use of the ***means-end*** and ***contribution links***. For instance, let us make us focus on a make contribution example (Fig.1).

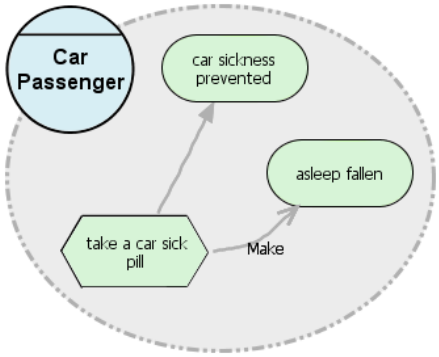


Fig. 1. The case of the passenger (Means-end link vs. Make Contribution link)

In it, a Car Passenger agent executes the take a car sick pill task in order to prevent himself from being sick during the journey he is making (Means-end link to car sickness prevented goal). As a side effect of this medication, the Car Passenger also goes to sleep (Make Contribution link to asleep fallen goal). Let us now carefully analyze this example. Both goals depicted in the model are equally accomplished: following the proposal in [8], we here assume that the ***means-end link*** leads to full accomplishment and the ***contribution link*** value is Make, which also indicates the goal is completely fulfilled. So then, what is the distinction among these two links? Do the ***means-end link*** and the ***make contribution link*** have the same semantics? If so, it would be best that the *i** framework only offered one of them to avoid construct redundancy [8], which may undermine the understanding and proper use of the language. However, it is our claim that these links are not the same. The difference here is given by the `Intention` behind the execution of the task.

As result of the mapping from i* tasks into UFO `actions`, every task is associated with a causing `intention` whose `propositional content` is a `goal`. In other words, we execute a particular task in order to accomplish a specific goal. In *i**, the association between the task and the goal in this case is made by a ***means-end link*** (e.g. take a car sick pill task as means to car sickness prevented goal). On the other hand, this same task can also generate some other goals to be accomplished, without however, being intended by the choice of this particular task. In this case, a ***make contribution link*** is established (e.g. take a car sick pill task as means to asleep fallen goal). By using the notion of *deliberately-achieves* previously defined, we may also make clear the distinction between the ***means-end link*** and the ***make contribution link*** (MakeCont).

$$action(a) \wedge goal(G) \wedge MakeCont(a, G) \rightarrow achieves(a, G) \wedge \neg deliberately\text{-}achieves(a, G)$$

As can be noted by the above definition in comparison with the means-end definition described a few paragraphs before, the only distinction between the two links is given by the causing intention. Table 2 summarizes some guidelines resulting from the analyses presented in this subsection for the use of the OR-decomposition relation, means-end link and make contribution link.

Table 2. Guidelines: OR-Decomposition, Means-end link and Make Contribution link

Hardgoals G ***---OR-decompostion-→*** hardgoals G1,G2 for an actor A iff 1. By accomplishing either G1or G2, G is accomplished
Action *a* ***---means-end-→*** hardgoal G for an actor A iff 1. By choosing to perform *a*, it was A's intention to achieve goal G, 2. Performing *a* causes situation S and 3. Situation S satisfies G
Action a ***---make contribution-→*** hardgoal G for an actor A iff 1. By choosing to perform a, it was NOT A's intention to achieve goal G, 2. Performing a causes situation S and 3. Situation S satisfies G

For reasons of lack of space, we refrain ourselves from presenting the results of the analysis of the break, help and hurt contribution links.

3 Conclusions and Future Work

In this paper, we have presented the long-term objectives and main results of our effort to provide an ontological foundation to the *i** language core. Our work so far has been conceptual, providing a clear formalizations to ensure that the semantics of the *i** constructs can be well understood and properly used. This work is not finished because i* is a powerful language allowing the expression of actor's theories about effects, side-effects and obstacles of actions. However, to move forward, we now need to do empirical work to validate the usability of our ontology, measured in for

example ease of learning and effort to use, and to validate the utility of the ontology for particular stakeholder purposes.

Acknowledgements

This work has been partially funded by the ProS-Req Spanish project (ref. TIN2010-19130-C02-00). We wish to thank the reviewers for their very constructive comments. We are also grateful to the support provided by FAPES (PRONEX #52272362/2010) and CNPq Productivity Grant #311578/2011-0).

References

1. López, L., Franch, X., Marco, J.: Making Explicit some Implicit *i** Language Decisions. In: 30th International Conference on Conceptual Modeling (ER'11), Berlin, Springer, LNCS, vol. 6998, 62--67 (2011)
2. Cares C., Franch X.: A Metamodelling Approach for *i** Model Translations. In: 23rd International Conference on Advanced Information Systems Engineering (CAiSE'11), Springer, LNCS, vol. 6741, 337--351 (2011)
3. Cares, C., Franch, X., Mayol, E., Quer, C.: A Reference Model for *i* In: Yu, E., Giorgini, P., Maiden, N., Mylopoulos, J. (Eds.). *Social Modeling for Requirements Engineering*, pp. 573-606. MIT Press, Cambridge, MA (2011).
4. Susi, A., Perini, A., Mylopoulos, J. and Giorgini, P.: The Tropos Metamodel and its Use, *Informatica*, 29, 401-408 (2007)
5. Cares, C.: From the *i** Diversity to a Common Interoperability Framework. PhD Thesis, Universitat Politècnica de Catalunya, Spain (2012)
6. Wachsmuth G.: Metamodel Adaptation and Model Co-adaptation. LNCS, vol. 4609, pp. 600--624 (2007)
7. Cares C., Franch X., Perini A. and Susi A.: Towards *i** Interoperability using iStarML. *Computer Standards and Interfaces*, 33, 69--79 (2010)
8. Guizzardi, R., Franch, X., Guizzardi, G.: Applying a Foundational Ontology to Analyze Means-End Links in the *i** Framework. In: 6th IEEE International Conference on Research Challenges in Information Science (RCIS'12), pp. 333-343. IEEE Press (2012).
9. Guizzardi, R., Guizzardi, G.: Ontology-based Transformation Framework from Tropos to AORML. In: Yu, E., Giorgini, P., Maiden, N., Mylopoulos, J. (Eds.). *Social Modeling for Requirements Engineering*, pp. 547-570. MIT Press, Cambridge, MA (2011).
10. Franch, X., Guizzardi, R., Guizzardi, G. and Lopez, L.: Ontological Analysis of Means-End Links. In *5th International iStar Workshop*, 37-42 (2011)
11. Franch, X.: Fostering the Adoption of *i** by Practitioners: Some Challenges and Research Directions. In: *Intentional Perspectives on Information Systems Engineering*, Springer, 177--194 (2010).

Using i* to Elicit and Model Transparency in the Presence of Other Non-Functional Requirements: A Position Paper

Luiz Marcio Cysneiros

School of Information Technology – York University, Canada
cysneiro@yorku.ca

Keywords: Software Transparency, Non-Functional Requirements, istar Framework

Abstract. Transparency has been, for long, a general requirement for democratic societies. The right to be informed as well as to have access to information has been an important issue on modern societies. Nowadays, Transparency has been elevated to a must have property to be delivered by governments and businesses. In an era when computer systems are ubiquitous and present in almost every aspect of our lives, it seems natural that Transparency becomes a key requirement in our software systems. We believe that Transparency can rarely be satisfied. The best we can do is to satisfy it within acceptable limits (satisfice). Therefore, we consider Transparency a Non-Functional Requirements (NFR) that should be elicited and modelled in the presence of other competing and synergistic NFR. Intentions behind the adoption of Transparency may play an important role while eliciting solutions for software to deliver appropriate levels of Transparency. Hence, we believe that the i* framework is ideal to elicit and model Transparency. This work will show initial ideas for using i* to support this effort.

1 Introduction.

Since 2005 we have been following the crescent number of claims from society for government and business to be transparent. In late 2007, the world faced the sub-prime mortgage crisis along with scandals due to enormous bonus for bankers involved in the administration of banks and industries that had to be bailed out by the US and Canadian government. This crisis, partially due to the lack of transparency in business and government regulatory schemas has contaminated other economies throughout the world. It all raised the cry for transparency to reach unprecedented levels.

In 2008, Canadians and Americans citizens were surprised to learn that when accessing Ticketmaster.com looking for tickets that were sold out, Tickemaster would alert they were being redirected to another site. However, it failed to alert that these customers were in fact facing ticket prices almost twice higher than in Ticketmaster. It called our attention that despite the fact that Ticketmaster was being transparent in its action it was not being transparent in its intentions. Driven by this episode, in 2009 we started to investigate the relationship between trust and transparency [2]. Contrary to what we expected, we could find many situations where trust would have a negative impact on transparency and vice versa.

We subscribe to the idea that in the near future the relationship among companies, clients, shareholders and government will be close to what Tapscott and Ticol [10] call "Naked Corporation": "Transparency must now be an explicit factor in nearly every management decision. Customers evaluate the worth of products and services at levels

never before possible. Employees share formerly secret information about corporate strategy, management behavior, and challenges. In a real-time global supply chain, companies and their business partners by necessity share competitive and operational secrets. With the rise of institutional investors, especially pension funds, share owners now scrutinize management like never before. Thanks to instant communications, *whistleblowers and inquisitive media, citizens and communities routinely put firms* under the microscope. And the Internet is a central locus and organizing force for all these activities".

Since software systems are now inherent to our society and part of our daily activities, it seems natural that such move will have to be followed by a change in our focus regarding software development. Systems will have to be developed since its early stages aiming at being Transparent. In fact, during the 2009 edition of the 17th IEEE International Requirements Engineering Conference, two Keynotes speakers[1] many times mentioned the importance of Software Transparency. Despite that, very little work has been done focusing this issue [1][6]

Many authors such as Holzner [5] and Henriques [4] provide different definitions for transparency but central to any definition remains the idea of *information disclosure.* We believe that for software to be Transparent the information which it deals with has to be transparent (Information Transparency). Software also needs to be Transparent itself. It should inform about itself, how it works, what it does and why (Process Transparency). Although approaches such as Open source and well defined and documented software development processes can contribute to software Transparency they alone do not come close to deliver Transparency.

We view Transparency as an NFR and hence, it may positively or negatively impact many other NFR and as a consequence it cannot be studied in isolation. Transparency should be treated as one of the most important contributing NFR since to satisfice Transparency it will be necessary to satisfice many other NFR hence, if transparency is not dealt as one of the leading NFR re-work may have to be done later to adapt existing solutions for NFR such as those illustrated in Figure 1 to cope with Transparency needs.

Moreover, as we could see from our study between Transparency and Trust [2], we need not only to model how, where and when systems need to be Transparent but also the intentions behind the intended Transparency. Every company *will have its own motivation to be transparent* and will deliver Transparency up to a point where it does not threat its business. Therefore, *modeling these intentions would help to clarify alternatives and to better reason about possible tradeoffs* during the elicitation and modeling process. We understand that *the i* framework provides an ideal environment* to support this process.

This work will show initial ideas on the use of i* to support eliciting and modeling Transparency requirements together with other NFR. We present a brief real life case study where Interoperability was studied together with Transparency as a proof of concept on how eliciting and modeling NFR correlations to Transparency may be critical to adequately achieve Transparency. Aside from that, we present other areas where we will investigate different issues to help developing software that will deliver Transparency such as which information should be kept, how it should be stored, how it should be retrieved, how it should be presented.

[1] Jim Herbsleb and Daryl Plummer

2 Objectives of the Research

Our short term goal is to better understand what does it mean to be Transparent and which information should be available to stakeholders to deliver Transparency without ignoring other NFR such as privacy, security, trust, and ethics among others. Note that Leite's work [6] heavily focus on transparency itself while in our focus the relationship between transparency and other NFR such as Trust, Privacy, Security, Ethics and Interoperability will concentrate the majority of our work.

We initially plan to capture this knowledge using catalogues in the form of Softgoal Interdependency Graphs (SIG) the same way we have been doing for capturing knowledge regarding NFR operationalizations in the past. However, we plan to further develop recent work aiming at providing a more efficient and flexible way to store and retrieve information gathered about Transparency and other NFR using ontology and semantic web techniques.

We also plan to investigate how we can categorize software regarding how much Transparency it delivers. We expect that such categorization will lead to standards to measure how Transparent one software is. Such classification may influence the decision process of acquiring one software.

Finally, we will investigate how to incorporate the knowledge for delivering Transparency with the process of eliciting and modeling software allowing the intentions behind adopting Transparencies solutions to be studied in conjunction with all the other intentional elements of the system being developed.

3 Scientific Contributions

Although Process Transparency may complement our research objectives, our main focus will be to investigate methods, techniques and tools to help requirements engineers to develop systems having Information Transparency in its core from the early stages of software development. We believe that the i* Framework will be ideal to provide the necessary structures and constructs to reason about ways to operationalize Transparency and co-related NFR as well as to capture the intentionality that drives individual business and governments to deliver transparency. I* intentional elements and its mechanisms to model intentions, its impacts and alternatives are ideal to deal with Transparency. Furthermore, modeling NFR plays a major role in i* and therefore is a perfect match to model Transparency where we will mostly be dealing with many NFR working together and intentions motivation companies to be and to offer transparency.

4 Ongoing and Future work

4.1 Investigating Transparency in the presence of other NFR

An initial approach to capture Transparency operationalizations can be found in the Transparency Ladder proposed by Leite [6]. In this Ladder we can see that central to deliver Transparency we can find other NFR, namely: Accessibility, Usability, Informativiness, Understandability and Auditability. Accessibility can then be decomposed into Portability, Availability, Operability and so on. Figure 1 Shows part of this Ladder. Of course any operationalizations that we may choose to satisfice Accessibility might contribute positively or negatively to operationalizations that can satisfice other NFR on the

Ladder. Moreover, other NFR outside the Ladder can also bring synergy or conflict to Transparency and vice versa. Hence, the challenge relies not only in finding out operationalizations to deliver Transparency but most importantly, how to harmonize these operationalizations in light of other much needed NFR. Initially, we will be registering our findings in catalogs using the NFR framework.

As a proof of concept, we have recently carried out *a case study* to evaluate the relationship between *Transparency and Interoperability*. This case study investigated particular types of messages exchanged between non-interoperable systems using a message broker within a Health Care Provider in Ontario. The model chosen is based on descriptive methods, particularly an observational case study used to analyze production data, other than laboratorial data, to create a baseline that may be used to verify if Interoperability can hurt or help Transparency. We used the Transparency Ladder proposed by Leite [6] to evaluate its proposed operationalizations for Transparency against possible problems brought by Interoperability mismatches. Figure 1 shows a *partial set* of the correlations that we found. The focus of this work was to evaluate how much incorrect data due to Interoperability issues occurred involving a system A and a system B after they have sent information to a system C. Analyzing these errors we tried to evaluate if despite the errors, Transparency could yet be satisficed at least in a timely manner. As we can see in Figure 1 we realized that there were many situations where errors due to Interoperability problems could have hurt Transparency. Since System C is used as repository of information to be manipulated later, these errors did not pose any risk to the daily operation of Systems A and B. However they would impact Transparency in a short term basis.

Other NFR such as Trust, Privacy, Security and Ethics will soon be investigated to evaluate its relationship with Transparency. Although at least intially we will continue to use SIGs to capture this knowledge, we plan to further develop a recent work using ontology and semantic web to facilitate NFRs knowledge reuse [7][8]. Further details will

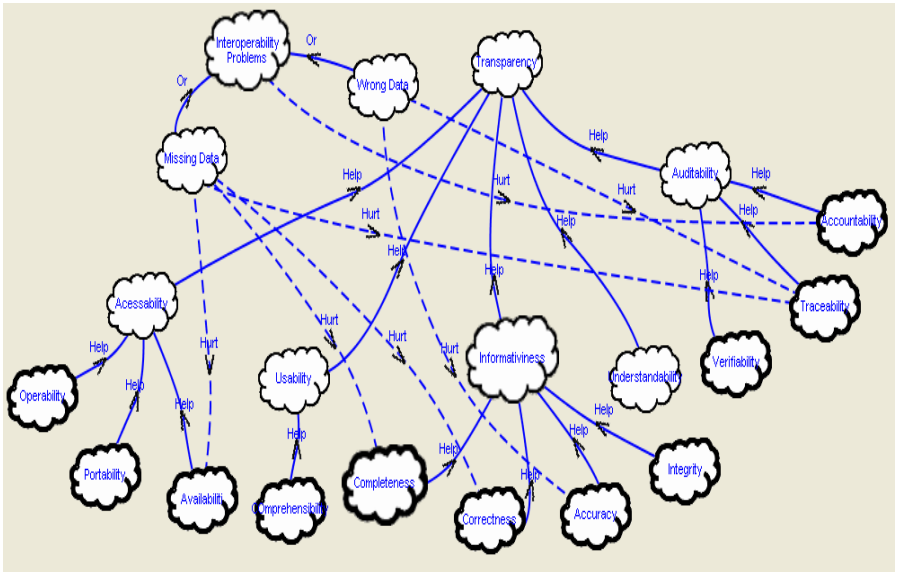


Figure 1 – Interoperability and Transparency

be shown in the section 4.2.

4.2 A Framework to Store and Retrieve knowledge on Software Transparency and Other NFR

SIGs have been used to represent quality attributes. Many complex projects can produce quite large and complex design graphs, which might complicate recovering alternatives, trade-offs and rationale information in SIGs. As an alternative, we started to look at ontologies and sematic web techniques as an alternative to manipulate NFR knowledge.

Ontologies are usually associated to description logic, and represented with languages like OWL [9], which provides vocabulary to describe classes, their properties, relations among them and cardinalities, and support inference rules to derive new information from input data. In operational terms, the Resource Description Framework (RDF) is widely used to describe ontologies (mainly at semantically enriched Web sites). We have developed the NFR and Design Rationale (NDR) Ontology where we implemented a separate faceted and multi-faceted search capability that can recover whole SIGs (or relevant SIG fragments) [8]. The NDR Ontology allows describing well-formed SIGs *in a machine-readable format* and processing the rationale knowledge embedded in SIGs by developing software applications. NDR describes well-formed SIG models: its classes represent SIGs concepts, and its properties describe feasible relationships among these concepts storing them in a central repository for future extraction. We are in the process of developing a tool that would semi-automate the storage process as well as provide query mechanisms to recover knowledge and, in a mid/long term basis, the design rational behind adopted solutions [7]. We expect that this tool would provide a way for retrieving Transparency and NFR related information both from a specific domain as well as from specific projects. This tool will be able to import from existing SIGs as well as to export to other Tools supporting SIGs manipulation. This repository may also be used to generate guidelines to classify software regarding its level of transparency as explained in Section 4.3

4.3 Supporting the Creation of Guidelines to Measure Software Transparency

We also aim at investigating the need for configuring software to deliver different levels of Transparency depending on the stakeholder using it. We anticipate that there will be situations where the levels of Transparency made available to some of the upper management people should not be available to every stakeholder for the sake of, among other reasons, having business strategies kept to those who need to know. Transparency may still have to be provided but with lower levels of Information Disclosure,

Based on the knowledge obtained during the previous phases of our research plan, we will gear towards a product-oriented approach to NFR classification to produce guidelines to be used by a semi-automated tool to attest how Transparent one software is. We will investigate how to create a tool that will support guided interaction with stakeholders that want to use a semi-automated approach based on this knowledge to classify one specific software. Different levels of Transparency will be achievable in a similar way one appliance can be classified for energy efficiency such as in the Directive 2010/30/EU. The use of GRL will be investigated as an option for providing weights schemas to help with the task of categorization. Standards based on these levels of

Transparency may guide software acquisition and Request For Proposal for outsourced software development

4.4 Integrate Transparency Knowledge to Development Models

Orthogonally to the goals mentioned above, we will investigate how to integrate the knowledge stored in the form of SIGs and/or using our tool into i* models reflecting the system being developed.

We believe that intentionality will play a key role to develop software delivering Transparency in accordance not only with the business/government needs but also taking into consideration citizen's needs. We expect that many tradeoffs will take place while addressing Transparency through the viewpoint of citizens, business and government together. We will need to integrate knowledge about Transparency with models depicting the functionality, non-functionality and the intentions related to the domain in question. We will be investigating systematic methods and possibly tools to integrate this knowledge into i* models. We also plan to evaluate if GRL can offer a richer environment when considering jUCMNav [3]. Particularly, we will be focusing on a GRL feature where intentional elements in an actor can be attached to importance factors. These factors can be linked to satisfaction levels to guide the evaluation of the overall satisfaction of an actor. We believe this characteristic may play an important role when evaluating different intentions behind Transparency and other functional and non-functional requirements. Other tools will be also evaluated.

References.

1. Cappeli,C. and Leite, J.C.S.P. "Exploring Business Process Transparency" in Proc. Requirements Engineering Conference, pp:389-390, 2007
2. Cysneiros, L.M. "An Initial Analysis on How Software Transparency and Trust Influence Each Other" 12th Workshop on Requirements Engineering on July 2009, Chile.
3. GRL – ITU-T – User Requirements Notation (URN) – Language Definition, ITU-T Recommendation Z.151 (10/12). Geneza, Switzerland(2012); http://www.itu.int/rec/T-REC-Z.151/en
4. Henriques A., Corporate Truth The Limits to Transparency, EARTHSCAN, UK (2007).
5. Holzner B., Holzner L., Transparency in Global Change: The Vanguard of the Open Society. University of Pittsburgh Press; 1 edition (2006).
6. Leite, J. and Cappelli, C. Software Transparency. Business & Information Systems Engineering, 2, 3 (2010), 127-139. DOI=10.1007/s12599-010-0102-z.
7. López,C., Codocedo,V., Astudillo,H., Cysneiros,L.M.. "Bridging the Gap between Software Architecture Rationale Formalisms and Actual Architecture Documents: An Ontology- driven Approach**.**. Science of Computer Programming Journal. *Volume 77 Issue 1, January 2012. Pg 66-80r*
8. López,C., Cysneiros,L.M., Astudillo,H.. **"**NDR Ontology: Sharing and Reusing NFR and Design Rationale Knowledge". First International Workshop on Managing Requirements Knowledge, Spain 2008. Workshop proceedings in the IEEE Digital Library, MaRK'08, p. 1-10. DOI: http://doi.ieeecomputersociety.org/10.1109/MARK.2008.7
9. McGuinness,D.L., van Harmelen,F. "OWL web ontology language overview," W3C recommendation, http://www.w3.org/TR/2004/REC-owl-features-20040210/
10. Tapscott, D. and Ticoli, D. "The Naked Corporation" The Wall Street Journal Ocotber 15th 2003

Using i* for Transparent Pedagogy

Elizabeth Suescún Monsalve and Julio Cesar Sampaio do Prado Leite,

Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro,
Rua Marquês de São Vicente 225, Ed. Padre Leonel Franca 13o. andar,
Rio de Janeiro, Brasil
emonsalve@inf.puc-rio.br, http://www-di.inf.puc-rio.br/~julio/

Abstract. Pedagogy, "the principles and methods of instruction" (Wordnet), implies a relationship among actors playing the roles of teacher and student and has a direct impact on the students' learning performance. In the past, the teacher was a transmitter of knowledge and the student a passive receiver. Nowadays, students are encouraged to challenge, deepen and create their own knowledge and teachers are supposed to lead this process. For that reason, transparency emerges as an important concern that aims to enhance this relationship by improving student awareness about the process and the contents of learning. The purpose of this research is to address the potential of i* within pedagogy transparency. We discuss the role of i* models as providing transparency for a game-based learning (GBL) strategy.

Keywords. iStar, Transparency, Pedagogy

1 Introduction

According to [1] pedagogy is "*study of teaching methods, including the aims of education and the ways in which such goals can be achieved*.", or according to [2] "*Pedagogy is more than the accumulation of techniques and strategies: arranging a classroom, formulating questions, developing explanations, creating a curriculum. It is informed by a view of mind, of learning and learners, of the kind of knowledge that is valued and above all by the educational outcomes that are desired*." As such, it can also be seen as a relationship between actors who have interpersonal contacts aiming the transfer of knowledge. In a broad sense, this relationship has a direct impact on a students' academic performance. For that reason, transparency is important [10], as to enhance students' awareness and their commitment towards learning [3].

The purpose of this article is to explore the question: *what is the potential of i* as an enabler of pedagogy transparency? As such, we discuss the role of i*[5] models as providing transparency for a game-based learning (GBL) strategy, focusing on a specific setting: Software Engineering Education by means of a GBL strategy.* In particular, we narrow down the general question by a first investigation of a specific situation: SimulES-W [4] as the implementation of the game.

Proposals for GBL in Software Engineering are being reported by different researchers [9]. Although there are positive evaluations for GBL [15], there is still lack of evaluations on the effectiveness of these strategies in SE. Our first experimental results [14], based on the application of tests to different groups, those which played and those who did not, shows that SimulES-W has a positive pedagogic effect. On the other hand, we also have experience in opening the game for students as an open source project. This gave us an insight that the software that implements the game provides an extra leverage if students can understand it. Since we have been working on software transparency [13], we decided to explore the transparency of SimulES-W as way of implementing pedagogy transparency, enabling students to better understand the inner workings of the game.

SimulES-W is a digital game used to teach software engineering [4], this game allows the players to play in a collaborative way. During the development process of SimulES-W the approach used was to base the requirements on the representation of intentionality between players. The resulting models were used to generate the implementation and to show how the game works not only from a technical approach but also from the point of view of the actor's intentionality. Using i* models we aim to show students how the game works from a conceptual modeling standpoint.

2 Objectives of the research

The aim of our research is to explore pedagogy transparency in the context of GBL, using the SimulES-W game. Pedagogy transparency [10] is a new concept not yet fully developed. The general idea is that, if students are told of how they are being taught, this may work in their benefit as to gain more knowledge as they become more aware of the teaching process, and as such have a more effective learning. Given that we have explored the potential of i* towards more transparent models [11], by means of our transparency conceptual model (accessibility, usability, informativeness, understandability and auditability) [13], we conjecture that i* models maybe a way of providing pedagogy transparency. As such, i* models will be used as way to providing transparency for a game-based learning (GBL) strategy, in particular of its own internal workings. That is, not only the game is used to enhance learning, but the game itself will be disclosed to the students (users) to inform them of how it achieves its goals.

Through SimulES-W, we will explore how i* models could provide support to the non-functional requirement of transparency [11, 13] as to be a means to disclose the inner workings of the game. We used a strategy that derives i*models using the Intentional Requirements Engineering method (Eri*c) [7]. The strategy uses, as a starting point, a lexicon [6] describing the vocabulary of the application. Intentional models are later on mapped to a MVC based architecture and to the source code.

Although the general question is how i* could enhance pedagogy transparency, we will study the question within a particular case of a multi-player, collaborative game. Our evaluation will be based on testing students using two different groups: a) students with exposure to GBL, and b) students with exposure to GBL and the GBL i*

models. Note that our proposed protocol will evaluate the use of i* as an enhancer of pedagogy, by providing some level of transparency. We will not compare i* models with other types of models for the same task. Central to our evaluation will be the question of the transparent rationale as an incentive to more effective learning, by leveraging student's awareness.

3 Scientific contributions

3.1 SimulES-W

SimulES-W is an evolution of the Problems and Programmers (PnP) game [12]; it aims at teaching software engineering process in a collaborative way, where a player covers the role of software project manager and this player has to deal with: budget problems, software engineers employment, and building of artifacts, all of that within the requirements of the project. Moreover, the player has to submit problems to other players, adversaries, to damage their game. SimulES-W has different rounds where players execute their moves such as: Start, Concept and Manage problems, Actions (Build, Inspect or Correct artifacts and integrate artifacts into a module), and Submit product.

3.2 The Modeling Process

SimulES-W [4] was developed using ERi*c [7], a method which uses i* as the main modeling language. ERi*c has 6 parts, interconnected by a bus (requirements baseline) through which they interact. The parts are: goal and actor elicitation, SDsituations identification, goal modeling for each actor, rationale modeling for each actor, Sdsituations specification, and analysis of SD and SR models. Strategic Dependency Situations (Sdsituations) identifies goals arrangements interconnected in order to implement how goals should be composed to set context dependency situations. Figure 1 portrays the SDsituation for the SimulES-W, which shows each round of the game. The rounds are named: Play round to start, Play round to actions, Build artifacts, Inspect artifacts, Play round to concepts, Managing problems, Submit product and Integrate artifacts in a module. Also, Figure 1 illustrates the time ordering required between rounds. Each round has its corresponding SD and SR Diagrams; Figure 3 illustrates one of them. Figure 2 describes the different actors, agents, roles and positions involved in the game, as seen from the software, informing the different types of actors and their instantiations.

3.3 Mapping Heuristics

SimulES-W is based on a MVC (Model-View-Controller) pattern to separate the business logic, interface, and control. Similar to the work presented in [8], we have devised a way of mapping i* models to an MVC architectural level description, which is described in [4]. This architecture is then reflected in the game´s code.

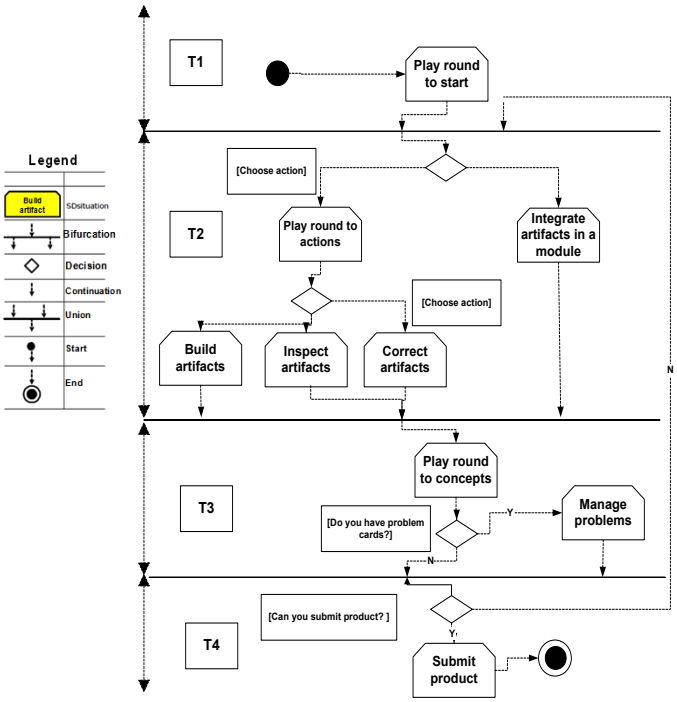


Figure 1. SDSituations Diagram [9].

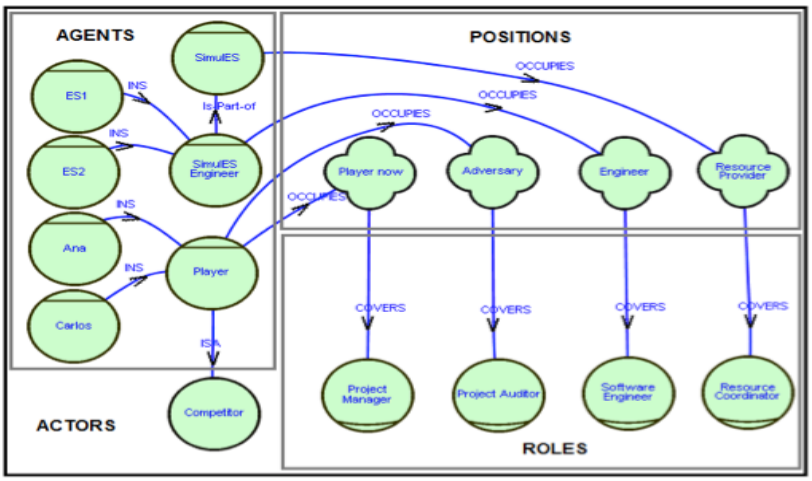


Figure 2. The SA Model for SimulES-W [9].

3.4 i* models as providing transparency for a game-based learning (GBL) strategy

If, in the process of helping learning, GBL becomes more transparent, we infer that the Pedagogy being used, will be more transparent; making the students more aware

about their learning process. As we can see from Figure 2, the reader of the model will be informed of the position "adversary" as being occupied by the agent "player", and that this position covers the role of "project auditor", and in Figure 3 the reader will be informed that the goal "project be accepted" depends on the position "adversary". In our transparency conceptual model [13], accessibility, is one of the qualities "helping" transparency. Providing access to the information (disclosure), via models, we are contributing to transparency, but, of course, the presence of other qualities will enhance transparency even more. The models produced are used as a way of showing how the game works, allowing the interested student to know how a GBL strategy is implemented. As such, the student will have access to how the pedagogy (GBL) is working. Section 2 described a general approach towards evaluation, based on tests. However, to better understand the results we have to consider levels of transparency (given that the concept is multi-faceted). As such, we will need a survey instrumented with questions to elicit the perceived level of transparency given our model [13], but also taking in consideration pedagogy [2, 3, 10].

4 Conclusions

We understand that transparent pedagogy involves characteristics as already mapped in [13], but we need to explore it further in the context of GBL. As we explore the frontier of transparent pedagogy we plan to continue to use i* models as base for the disclosure of information about the game and also regarding the context in which the learning takes places. Of course our models will evolve along the preparation for the experimental study, as we learn more about transparent pedagogy.

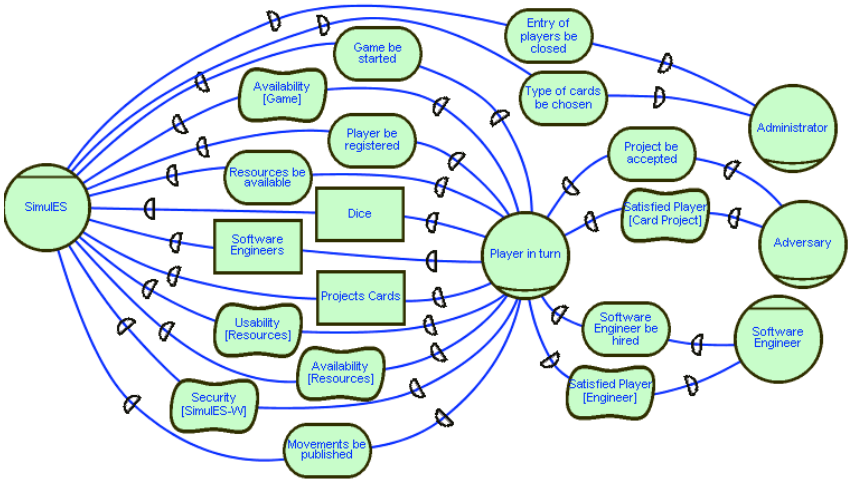


Figure 3. SDsituation: Play round to start.

5 Ongoing and future work

We are starting to understand transparent pedagogy in the context of GBL evaluation for software engineering. We will use a survey approach, which should blend transparency with more knowledge of pedagogy transparency. This work will stand upon early work on the game evaluation [9, 11], which uses both qualitative and quantitative questionnaires, in order to build an evaluation mechanism to understand the role of conceptual models in supporting a transparent pedagogy. As our results become available we will be in a better position as to infer the implications of the results towards the question of how intentional models may help pedagogy transparency.

References

1. Encyclopedia Britannica. Available at: http://global.britannica.com/EBchecked/topic/448410/pedagogy (April 2013)
2. Leach, J and Moon, B. Recreating pedagogy, Learners and pedagogy, 265–276. (1999)
3. Frank Coffield, David Moseley, Elaine Hall, Kathryn Ecclestone, Learning styles and pedagogy in post-16 learning A systematic and critical review. Published by the Learning and Skills Research Centre www.LSRC.ac.uk, (2004)
4. Monsalve, E. Construindo um Jogo Educacional com Modelagem Intencional Apoiado em Princípios de Transparência. Dissertação de Mestrado, PUC–Rio. (Março de 2010)
5. Yu, E. Modeling Strategic Relationships for Process Reengineering. Ph.D. Thesis, Graduate Dept. of Comp. Science, University of Toronto, (1995)
6. Leite, J.C.S.P. A Strategy for Conceptual Model Acquisition, Proceedings of IEEE International Symposium on Requirements Engineering, IEEE, San Diego, Ca, USA, pp. 243-246. (1993)
7. Oliveira, A. P. A. , leite, J.C.S.P., Cysneiros, L.M. Método ERi*c-engenharia de requisitos intencional11th Workshop on Requirements Engineering Barcelona, WER (2008)
8. Alencar F, Marín B, Giachetti G, Pastor O, Castro J AND Pimentel J. From i* Requirements Models to Conceptual Models of a Model Driven Development Process. The Practice of Enterprise Modeling; Second IFIP WG 8.1 Working Conference, PoEM 2009, Stockholm, Sweden, November 18-19, (2009)
9. Monsalve, E., Werneck, V. & Leite, J.C.S.P. Teaching Software Engineering with SimulESW. Proceedings of XXIV Conference on Software Engineering Education and Training (CSEE&T 2011), Hawaii, USA. (pp. 31–40). (2011)
10. Dalsgaard, C., & Paulsen, M. F. Transparency in Cooperative Online Education. The International Review of Research in Open and Distance Learning, The International Review of Research in Open and Distance Learning, 2009 (Vol 10, No 3). Retrieved from http://www.irrodl.org/index.php/irrodl/article/view/671/1267. (2009)
11. Leite, J.C.S.P.; Capelli, C. Exploring i* Characteristics that Support Software Transparency, in Proc. Of the 3rd International i* Workshop, CEUR Workshop Proceedings Volume 322, pp. 51-54. (2008)
12. Baker, A. Problems and Programmers. Honors Thesis, Department of Informatics, School of Information and Computer Science, University of California, Irvine, CA. (2003)
13. Leite, JCS do Prado and Cappelli, Claudia Software Transparency, Business & Information Systems Engineering 2 (3), 127 – 139 (2010)
14. Monsalve, E., Werneck, V., Leite J. C. S. P., SimulES-W: A Collaborative Game for Teaching Software Engineering. Computers & Education, (Submitted April 2013)
15. Ebner, M. & Holzinger, A. Successful implementation of user-centered game based learning in higher education: an example from civil engineering. Computers and Education, 49(3), 873–890. (2007)

Uncertainty in Goal and Law Modeling and Analysis

Silvia Ingolfo, Jennifer Horkoff, John Mylopoulos

University of Trento, Italy

Abstract. Goal models are widely recognized as an effective means for capturing requirements for socio-technical systems. Recently, models of law have been investigated and analyzed in conjunction with goal models, in order to evaluate the legal compliance of software system requirements. As goal models capture social, often ill-defined concepts, and as law models capture ambiguous legal settings, both models are characterized by the presence of uncertainty. Consequently, both goal and law models consider uncertainty as part of their analysis, allowing for unknown or inconclusive analysis labels. However, it is also possible to consider uncertainty in the content of such models. Recent work has applied an existing formal method for capturing uncertainty in goal models. In this paper we make a distinction between uncertainty in analysis and uncertainty in content, reporting on the influence of such uncertainty in models of law and requirements.

1 Introduction and Objectives

The usefulness of goal models (such as i* [1]) in capturing socio-technical requirements is widely recognized in Requirement Engineering. The impact of the law in both functional and non-functional requirements has gained a lot of attention in recent years. Software that is not designed in compliance with applicable laws can cause great economic damage to organizations. To limit such outcomes, it has become imperative to establish of a software system as early as the requirement phase. So on one side we have goal models for representing requirements, and on the other we want to represent and model law. The Nòmos 2 framework [2], inspired by RE ideas, models laws in terms of *norms* and *situations*. The link between these models provides a previously missing step toward the evaluation of regulatory compliance of a requirements model [3].

As goal models capture early, social requirements, uncertainty is an unavoidable factor that has not been widely investigated. Uncertainty is also present in laws, arising from the intricate structure of law, as well as ambiguities and exceptions. Although law models (e.g., Nòmos 2 [4]) can take legal variation into account, they cannot easily express uncertainty over these variations, or exploit uncertainty information as part of analysis.

In this paper we cover two categories of uncertainty: (1) uncertainty in analysis results and (2) uncertainty captured in the model structure. The first type of analysis has been explicitly considered for both goal and law models. Recent

work has considered (2), explicitly capturing uncertainty over the structure of goal models [5]. We consider the application of these ideas to law models, and outline future work which may combine (1) and (2) for goals and/or laws.

Objectives: In this paper we discuss the explicit consideration of uncertainty in both goal and law modeling and analysis, exploiting the synergies of existing work, and outlining new avenues of uncertainty-related investigation.

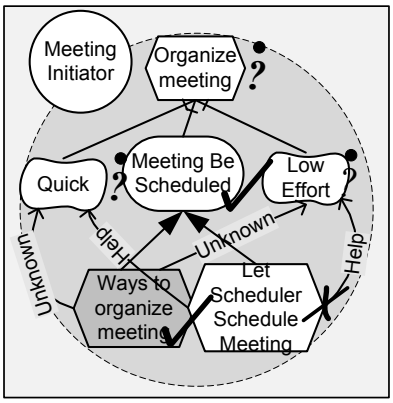
2 Background: Nòmos 2

Nòmos 2 [4,6] is a modeling framework for representing law. The concept of Norm is defined as a 5-tuple (*type*, *holder*, *counterpart*, *antecedent*, *consequent*). *Type* is the type of the norm (e.g., duty or right). *Holder* is the role that has to satisfy the norm, while the *counterpart* is the role whose interests are helped if the norm is satisfied. *Antecedent* and *consequent* are modeled in terms of situations and they represent the conditions to satisfy to make the norm applicable (antecedent) and the conditions to satisfy in order to comply with the norm (consequent). A situation is defined as a partial state of the world – or state-of-affairs – represented as a proposition which can be true, false, or have an unknown truth value.

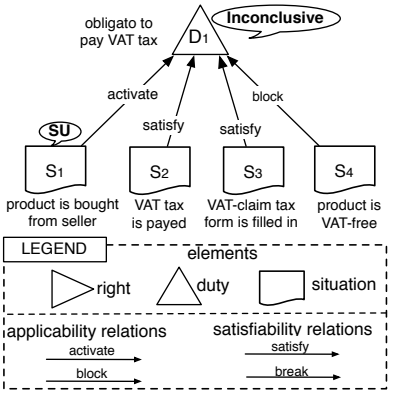
The idea behind Nòmos 2 is that a set of situations make a norm applicable and similarly situations can satisfy the norm. To capture this applicability and satisfiability, we model the relations between situations and norms as label propagation mechanism. The two relations for satisfiability ($satisfy/break$ propagate positive/negative satisfiability) and two relations for applicability ($activate/block$ propagate positive/negative applicability) link situations to norms. In Nòmos 2 situations are propositions that can be known to have Satisfiability True (ST), False (SF), or Unknown (SU). Similar label are propagated by the relations for Applicability (True AT, False AF, Unknown AU). Depending on the satisfiability of the input situations, the target norm receives true/false/unknown values for satisfiability and applicability. The combination of this two values defines the compliance value for a norm (compliant, not-compliant, tolerated, or inconclusive). Composite relations (*derogate*, *endorse*, *imply*) capture the relations between norms [4]. In figure 1b we show an example of a Nòmos 2 model representing a simplified norm about VAT-tax.[1]

When a product is bought (sat(s_1)=ST), the relation $s_1 \xrightarrow{\text{activate}} D_1$ propagates positive applicability to the norm. When the situation s_2 is also satisfied, then the relation $s_2 \xrightarrow{\text{satisfy}} D_1$ propagates positive satisfiability, and we say that the duty is complied with (it is applicable and satisfied). Propagation for s_3 is similar. However when s_4 holds (the product is VAT-free), then the relation $s_4 \xrightarrow{\text{block}} D_1$ propagates negative applicability (label 'AF') and the duty is not applicable.

[1] The graphical notation used to express the label is only used for illustrative purpose.



(a) Example i* Analysis over a Subset of a Meeting Scheduler Example



(b) Example of a Nòmos 2 model for the duty to pay VAT-tax on product.

Fig. 1: Analysis Examples Showing Uncertain Analysis Results

3 Scientific Contributions

We make the distinction between uncertainty in analysis results and uncertainty over the structure of the model. In this paper, the former refers to uncertainty about the satisfaction or applicability of a particular model element, while the latter refers to uncertainty about the presence, uniqueness, or number of model elements and links. We illustrate this distinction in the following.

3.1 Uncertain Analysis Results

Goal Models. Goal models have long provided a "lightweight" consideration of uncertain analysis results using the *unknown* contribution link and *unknown* analysis value ($\overset{\bullet}{?}$), with the former intended to represent a contribution with an unknown type (e.g., help, break), and the latter meant to represent the presence of evidence with unknown polarity (satisfied/denied) and strength (full/partial) [7,8]. For example, in Figure 1a, part of a simple meeting scheduler example, we propagate initial satisfied and denied labels through two unknown contribution links, producing unknown analysis labels for Quick, Low Effort, and ultimately for Organize meeting.

Nòmos 2 Models. On the legal side, a Nòmos 2 model allows us to express and reason about the uncertainty related to the situations holding, as well as the consequences this uncertainty has on the compliance of the model. The analysis of these models can therefore explore how the uncertainty in the situations holding (e.g., domain assumptions or hypothetical scenario) affect the compliance with applicable laws. For example in the scenario where it is unknown whether a product is bought (sat(s_1)=SU), then the applicability of the norm is unknown because the relation $s_1 \xrightarrow{\text{activate}} D_1$ propagates unknown applicability. In

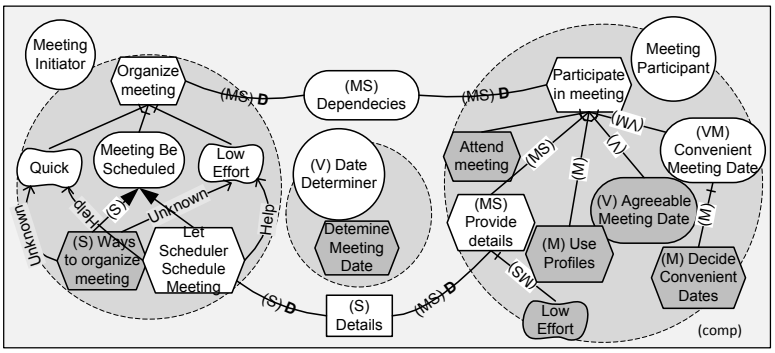


Fig. 2: Uncertainty Annotations in an i* Model (from [5])

Nòmos 2 the norm is evaluated to `inconclusive`: when it is not known whether the norm applies or not, it is not possible to infer any conclusions about it.

Discussion. Both goal and law models allow for unknown analysis values as initial values/assumptions, starting analysis. Such uncertainty is propagated using existing reasoning procedures, as described. Unlike Nòmos 2 models, goal models contain a simple form of uncertainty in the type of contribution relation. We explore this type of uncertainty — uncertainty over model structure — in the next section.

3.2 Model Uncertainties

Goal Models. Previous consideration of uncertainty in the structure or contents of goal models was limited only to uncertainty in contribution links (unknown). Although useful, uncertainty may occur in any relationship or element. Recent work has used the *MAVO* formal uncertainty framework [9] in order to capture uncertainty in a more general and expressive form. In this approach, we limit our focus to possibilistic uncertainty, as opposed to probabilistic uncertainty.

MAVO is a language-independent approach for formally expressing uncertainty in models. It allows users to express uncertainty using a set of annotations over elements and relationships in their model. As these annotations can be applied to any type of model (any metamodel), the approach is language independent. Specifically, the approach allows for annotations M, V, and S over model elements and links, and COMP (complete) or INC (incomplete) for the entire model. We illustrate application of this framework to i* in Figure 2 adapted from [5].

The M (May) annotation allows us to express uncertainty about the presence of an element or link in a model. In our model, we are uncertain, for example, about whether we really need to Use Profiles as part of Participate in Meeting. The V (variable) annotation allows use to express uncertainty about element distinctness. We are uncertain if Agreeable Meeting Date and Convenient Meeting Dates are distinct goals, or could be merged. The S (set) annotation represents uncertainty about the number of elements, elements which may be sets. We know we must provide Details, but are not sure if there is one detail, or many, or what

those details are. We mark the entire model as COMP, meaning that there should be no more new elements or relations.

MAVO captures uncertainty formally by expressing metamodels and constraints in First Order Logic, removing constraints which ensure the presence, distinctness or number of each element in the formalism. This allows use of existing solvers to find "solutions", corresponding to concrete, uncertainty-reduced models. More detail can be found in [9,5].

Nòmos 2 Models. As Nòmos 2 models are also characterized by uncertainty, such a general uncertainty framework could be applicable. The possibility to express uncertainty in the structure of a Nòmos 2 model, could be useful when sources are uncertain. For example we could annotate the fact that a situation "product bought at the airport" May block (make not applicable) the duty to pay the VAT-tax. The uncertainty related to this annotation arises because not all airport products are tax-free: the ones bought at the duty-free are, but products at regular shops usually include VAT. Similarly we could annotate that we are uncertain whether it is enough to fill in the VAT-claim form or maybe there is something else to provide in order to be really compliant. For example, when submitting these VAT-claim forms at the custom office, some identification documents are needed for the passenger. However, it could be that the proof a valid return ticket is also needed to really comply. We can model this using MAVO by adding additional S and M annotated situations (e.g., (M) valid return ticket is provided, (MS) passenger identification documents provided) which can satisfy the norm. Further investigation and examples are needed to evaluate the combination of Nòmos 2 and *MAVO*.

Discussion. In this section we have explored uncertainty over the structure of the model, while previous considerations of uncertainty assumed that the model was certain but considered uncertainty in analysis values. In some cases, the border between uncertainty in model structure and analysis results is difficult to define. We provide a preliminary sketch of these dimensions in Figure 3. Existing approaches for goal models consider only a small amount of model uncertainty (unknown links) and uncertainty in analysis using the unknown label (point (1) in Figure 3). Nòmos 2 considers uncertainty in satisfaction and applicability as part of analysis, but does not consider uncertainty over the model (point (2)).

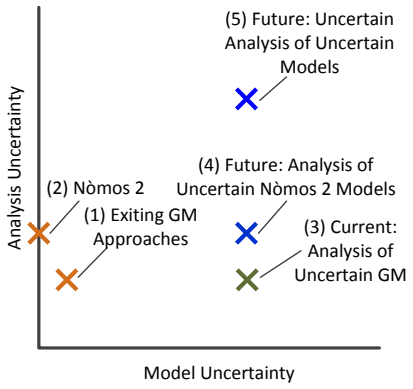


Fig. 3: Model vs. Analysis Uncertainty - Existing and Proposed Approaches.

We can envision investigation using other combinations of these dimensions. We are currently investigating ways to apply i* analysis over *MAVO*-annotated i* models (point (3)). This work would allow us to explore analysis results given possible uncertainty reductions, in order to explore alternative requirements by producing sets of possible labels even before uncertain-

ties are resolved. Similar approaches could integrate the semantics of Nòmos 2 models with *MAVO* annotations, considering uncertainty in the evaluation of compliance (point (4)).

By analyzing uncertain goal models, we increase our consideration of model uncertainty (beyond unknown links), but do not consider any further uncertainty in analysis results (beyond the use of unknown labels). Future work could support analysis which allows for the possibility of more uncertainty over possibly uncertain models (point (5)). For example, if we explicitly consider the effects of an open world assumption (INCOMP) during model analysis, the possibility of additional elements and links may cause further types of uncertainty in our analysis results (e.g., an element may be satisfied). Our definition of model vs. analysis uncertainty may evolve as we consider such possibilities.

4 Conclusions and Future Work

In this paper we make the distinction between uncertainty over the contents of the model (model uncertainty) and uncertain analysis results (analysis uncertainty). We have summarized uncertainty as considered in the analysis of i* and Nòmos 2 models. We have summarized existing work on model uncertainty for goal models, and provided examples of how such work can be applied to Nòmos 2 models. We outline possible future work considering other combinations of uncertainty in modeling or analysis. In the future, we may need further distinctions to characterize uncertainty, e.g. design-time uncertainty over model contents vs. run-time uncertainty over unpredictable aspects of the environment.

References

1. E. Yu, "Towards modelling and reasoning support for early-phase requirements engineering," in *Proceedings of 3rd IEEE International Symposium on Requirements Engineering*, vol. 97, pp. 226–235, IEEE, 1997.
2. A. Siena, S. Ingolfo, A. Susi, I. Jureta, A. Perini, and J. Mylopoulos, "Requirements, intentions, goals and applicable norms," in *RIGIM - ER Workshops*, p. 195, 2012.
3. S. Ingolfo, A. Siena, and J. Mylopoulos, "Establishing regulatory compliance for software requirements," in *Conceptual Modeling - ER 2011*, vol. 6998 of *Lecture Notes in Computer Science*, pp. 47–61, 2011.
4. A. Siena, I. Jureta, S. Ingolfo, A. Susi, A. Perini, and J. Mylopoulos, "Capturing variability of law with Nòmos 2," in *Conceptual Modeling - ER 2012*, vol. 7532 of *Lecture Notes in Computer Science*, pp. 383–396, 2012.
5. R. Salay, M. Chechik, and J. Horkoff, "Managing Requirements Uncertainty with Partial Models," in *Proc. of RE'12*, 2012.
6. S. Ingolfo, A. Siena, I. Jureta, A. Susi, A. Perini, and J. Mylopoulos, "Choosing compliance solutions through stakeholder preferences," in *REFSQ 2013*, vol. 7830 of *Lecture Notes in Computer Science*, pp. 206–220, 2013.
7. L. Chung, B. Nixon, E. Yu, and J. Mylopoulos, *Non-Functional Requirements in Software Engineering*. Kluwer Academic Publishers, 2000.
8. J. Horkoff and E. Yu, "Interactive Analysis of Agent-Goal Models in Enterprise Modeling," *Int. J. Inf. Syst. Model. Des. (IJISMD)*, vol. 1, no. 4, pp. 1–23, 2010.
9. R. Salay, M. Famelis, and M. Chechik, "Language Independent Refinement Using Partial Modeling," in *Proc. of FASE'12*, vol. 7212 of *LNCS*, pp. 224–239, 2012.

Qualitative vs. quantitative contribution labels in goal models: setting an experimental agenda

Sotirios Liaskos Saeideh Hamidi Rina Jalman

School of Information Technology,
York University
Toronto, Canada
{liaskos,hamidi,rjalman}@yorku.ca

Abstract. One of the most useful features of goal models of the i^* family is their ability to represent and reason about satisfaction influence of one goal to another. This is done through contribution links, which represent how satisfaction or denial of the origin of the link constitutes evidence of satisfaction/denial of its destination. Typically in the i^* family, the nature and level of contribution is represented through qualitative labels ("+", "−", "++" etc.), with the possibility of alternatively using numeric values, as per various proposals in the literature. Obviously, our intuition seems to suggest, labels are easier to comprehend and to come up with, while the use of numbers raises the question of where they come from and what they mean, adds unwarranted precision and overwhelms readers. But are such claims fair? Based on some early experimental results, we make the case for more empirical work on the matter in order to better clarify the differences and understand how to use contribution representations more effectively.

Key words: requirements engineering, goal modeling, i-star

1 Introduction

Goal models of the i^* family [11, 2, 4] have been found to be useful for qualitatively representing and analyzing how stakeholder goals influence each other. The concept of the *contribution* link is used to show how satisfaction or denial of the goal which is origin to the contribution affects satisfaction/denial of the goal that is targeted by the contribution. The level of contribution, i.e. how strong the influence is, is represented with a label that decorates the link. Typically, this label is a symbol such as "+", "−−", "++" etc. denoting both whether the contribution is positive or negative and offering a coarse characterization of the strength of contribution. However, the use of numerical labelling has also been proposed [4, 8, 1]. Such labels may be simple numeric instantiations of the same modeling principles (e.g. [4]) or can have quite more distinct semantics from their qualitative counterparts [8, 1], implying also different ways of inference about satisfaction influence.

Which type of label should we then use? Our intuition suggests that qualitative labels are easier to comprehend and to come up with, while the use of

numbers automatically raises the question of where they come from [7, 3], adds unjustified precision and discourages readers. Is that true? In this paper, we review two of our exploratory experimental studies that seem to suggest that use of numbers is not to be dismissed. In both studies, participants are asked to perform ad-hoc reasoning about optimal solutions by just looking at different goal graphs. The results do not offer any evidence that numerical representation obstructs success in that task; instead participants seem to be able to find the optimal solution using the numbers. Using these studies we make a case for more experimentation on the subject, in order to not only learn more about the visual properties of goal modeling languages but also force ourselves explicate what exactly we intend those visual languages to be used for.

The rest of the presentation is organized as follows. In Section 2 we discuss qualitative versus quantitative contribution in goal models. In Section 3 we describe our experiments. Then in Sections 4 and 5 we offer conclusions and our plans for the future.

2 Objectives of Research

Modeling and reasoning about contribution in i^* languages is generally understood as something to be done in a qualitative fashion. The use of qualitative labels is rooted on the idea that Non-Functional Requirements (NFRs), the matter modeled through e.g. i^* soft-goals, need to be dealt with in a way that does not necessitate availability of accurate empirical data and complex and hard calculations of global optima [10]. Through qualitative analysis, rough assessment of goal satisfaction is possible by examining whether there is evidence of support for some satisfaction of a goal combined with lack of evidence against such satisfaction. This is sufficient to know that important NFRs are satisfied to a good enough degree. Thus, as the framework opts for rough characterizations of satisfaction and influence thereof rather than precise analysis of quantitative data from the field, using qualitative labels is a logical choice.

However, Giogrini et al. show that numbers can be used as well [4]. These numbers are not measurements from the field but simply numeric representations of satisfaction and contribution levels, otherwise presented with qualitative symbols. Similar attempts, which depart from the label propagation semantics have also been proposed in the literature as surveyed by Horkoff and Yu [6]. But the utilization of numbers in place of qualitative labels raises two important issues: (a) the question how numbers (with all their precision) are elicited and (b) the suspicion that "+" and "−" are easier to comprehend as contribution labels than, say, 0.9 and 0.3.

We have recently shown, however, that numeric representations of contributions have their merits [8]. With respect to question (a) above, i.e. where the numbers come from, making the assumption that the goal graph is acyclic and separated from a hard-goal AND/OR decomposition, we showed that the Analytic Hierarchy Process (AHP) can be used to elicit numeric contribution links – an idea also proposed earlier [9]. Simply, the soft-goal hierarchy is viewed as AHP

criteria hierarchy and each OR-decomposition is treated as a separate decision problem. As such, the optimal solutions that the graph yields can be argued to be as valid as the AHP decisions they correspond to. This, in turn, supports the relevance of the numeric labels themselves when used for that specific purpose (i.e. deciding optimal solutions).

What is perplexing, though, is concern (b) above: the use of the numbers not to just make the AHP decisions but also as contribution labels on the goal model in order to convey information to readers. Moreover, the question seems to extend to both qualitative and quantitative approaches. The issue seems to be the difficulty to define what exactly this information is, i.e. what exactly the reader is supposed to learn or understand by looking at models such as those of Fig. 1. Moreover, if we explicate the objective of the representation, it is logical to subsequently ask whether there are ways to read it (i.e. ways to understand labels and how they combine) that are more natural and effective than others. Below we describe our attempts to understand this problem better via conducting two small experimental studies.

3 Scientific Contributions

The information contained in the contribution structure (i.e. a portion of a goal model containing contribution links) amounts to a set of binary relations between goals showing how one contributes to the other. Of course, this information may as well be expressed in text or a catalogue of separate logical formulae. But we choose to put all these individual pieces in a graph, because we apparently aim at presenting a whole that emerges by combining them. In particular, if we consider contribution structures to be visual representations of a decision-making problem, we seem to be hoping that using a graph facilitates ad-hoc detection of good decisions. In other words, by just looking at such diagrams some readers must be able to intuitively combine individual contribution links and detect optimal solutions. Moreover, it may be fair to even assume that the detection process is natural and does not assume prior training to the language.

We investigated whether the current representations of contribution indeed have such properties and whether the choice of contribution labels (qualitative vs. numbers) has any influence [8]. We presented to ten (10) experimental participants small goal models (5 soft-goals, 11-15 contributions) with either quantitative or qualitative labels. We first constructed the quantitative ones using random AHP priority numbers. Then, to construct the qualitative ones, we replaced the numbers in the quantitative models with qualitative labels by discretizing the continuous interval [0,1]. So a value in [0,0.2) is replaced by "−−", a value in [0.2,0.4) is replaced by "−" etc. We presented the models to the participants in a within-subjects counterbalanced design and asked them to find for each model, without using any other aids, the optimal out of two/three options (children of OR-decompositions). We used the AHP definition of optimality in both cases. The participants are graduate students of Information Technology, and are not told anything about how to reason with goal models beforehand.

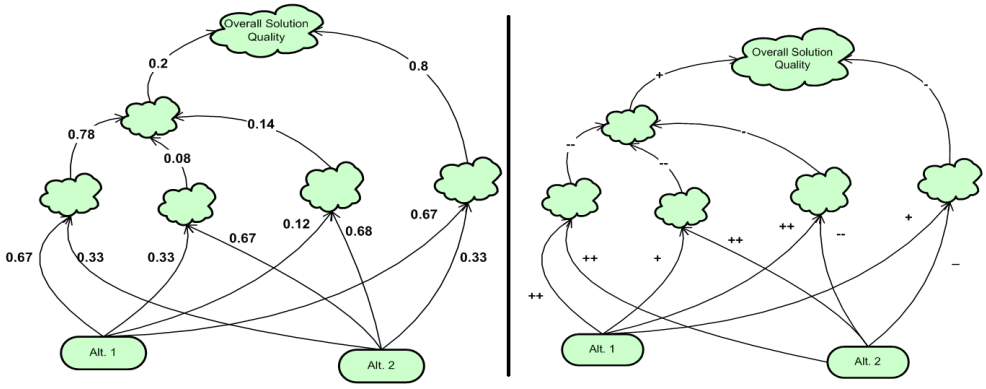


Fig. 1. Which solution has the best overall quality? (you have 60 seconds!)

In the quantitative case, the participants answered correctly in the majority of the times; the binomial test confirmed that this is not the result of randomness (0.95 significance). We might then be allowed to hypothesize that, for small models, the visual result of quantitatively labelling goal models allows readers to correctly guess the optimal (according to AHP) decision, by just looking at the model. Did use of qualitative labels ("+", "−" etc.) have an even greater effect? Our result was that the participants were not more or (significantly) less able to actually identify the correct (according to AHP again) alternative, compared to the quantitative case.

Inspired be these early indications, we recently performed a second study. This time we presented 10 models, five (5) quantitative and five (5) qualitative to another eight (8) participants. Each goal model of one group matches a model of the other group in all aspects except for the contribution labels, as seen in Fig. 1. For the labels, we randomly assign contribution symbols and values, to produce qualitative and quantitative models respectively. Overall, the models contain 4 to 7 soft-goals and 6 to 14 contribution links in various organizations and two alternatives to choose from. We present the models separately in a random order to the participants and we give them sixty (60) seconds to select the optimal alternative for each. The optimal solution in the qualitative models is this time based on the standard label propagation algorithm. Of the forty (40) answers received for each type of model, qualitative and quantitative, sixteen (16) and thirty (30) agree respectively with our calculation of the optimal. The binomial test shows that in the latter case (the quantitative responses), the result is significantly unlikely to be random. Further, while in the qualitative case the successful responses fluctuate with respect to model size (in a non-characterizable fashion), the successful responses in the quantitative case are unaffected by model size (6 out of 8 persons get it right for all models).

To sum up, considering contribution structures as visual instruments for ad-hoc detection of optimal solutions, these small studies aim at understanding whether there is an a-priori way by which uninitiated readers expect that such instruments work. The preliminary evidence appears to support that some par-

ticipants may have some prior inclination to deal with numbers in a specific way. This, we conjecture, might be due to the fact that dealing with weights, percentages and proportions is much more common in education and daily life than the use of custom symbols like qualitative labels. Regardless, we believe that until we have conclusive results we must resist the temptation to dismiss numerical representations of contribution as confusing or difficult to comprehend.

4 Conclusions

The possibility of qualitative satisfaction analysis is one of the major advantages of using goal modeling notations of the *i** family. However, when one considers goal models as visual instruments for making quick assessments about optimal decisions, the question of naturalness and efficiency of the representation arises. Our preliminary trials on readers without prior training seem to suggest that numeric representations evoke a way of visual reasoning that is consistent with simple aggregation arithmetic over contribution labels.

If further study confirms this or other similar results, the emerging research problem is how we incorporate them in the language itself, also in a way that both the good properties of qualitative reasoning are preserved and a systematic elicitation approach remains available. For example, designs that depart from static visual representations, such as interactive evaluation procedures [5, 3] seem to offer a possible answer to this problem by combining the elicitation and representation aspects. In any case, the more we focus on the role of the goal model as a communication and comprehension aid, the more relevant empirical work with ordinary readers becomes.

5 On-going and Future Work

The specific empirical goal we have set, i.e. that of understanding the comprehensibility of different labelling and aggregation strategies for contributions, requires us to consider many more experimental trials before getting a clear picture. Sizes, structures of models and domains they represent, numerical precision levels or even font sizes and shapes are examples of simple variables that need to be controlled for.

Furthermore, depending on what training and background assumptions one makes for the use of *i** representations in practice there are at least two ways to organize the investigation. One possibility is to keep looking for natural a-priori visual reasoning procedures, formed through other experiences of the reader or potentially utilized through analogies. In that case the educational, professional or other background of the reader becomes an important factor. We plan to run the same experimental procedures with student participants from non-mathematical disciplines such as humanities and from random samples from the general population. We plan to also add a qualitative component in order to elicit how participants exactly think whenever they try to reason about contribution structures. Are for example any analogies utilized, i.e. does the representation

remind them of something familiar to which they refer in applying a reasoning strategy? A second possibility is to assume that contribution labelling and aggregation models are a subject to be trained at beforehand. In that case, the learnability of these models needs to be tested, such as how soon and with what accuracy trainees are able to comprehend a contribution structure.

Finally, in all cases, comprehensibility criteria can be defined in different ways. For us it is the ad-hoc detection of optimal solutions. Alternative criteria include how quickly and accurately the user can read and remember individual contributions, successfully explain a given satisfaction degree or, say, detect conflicting nodes. By thinking of concrete criteria we actually force ourselves to think, decide and propose what uses of goal models are important and why. This call for reflection is, we find, one of the greatest benefits of experimental work.

References

1. Daniel Amyot, Sepideh Ghanavati, Jennifer Horkoff, Gunter Mussbacher, Liam Peyton, and Eric S. K. Yu. Evaluating goal models within the goal-oriented requirement language. *International Journal of Intelligent Systems*, 25(8):841–877, 2010.
2. Daniel Amyot and Gunter Mussbacher. User requirements notation: The first ten years, the next ten years. *Journal of Software (JSW)*, 6(5):747–768, 2011.
3. Golnaz Elahi and Eric S. K. Yu. Requirements trade-offs analysis in the absence of quantitative measures: a heuristic method. In *Proceedings of the ACM Symposium on Applied Computing (SAC'11)*, pages 651–658, Taichung, Taiwan, 2011.
4. Paolo Giorgini, John Mylopoulos, Eleonora Nicchiarelli, and Roberto Sebastiani. Reasoning with goal models. In *Proceedings of the 21st International Conference on Conceptual Modeling (ER'02)*, pages 167–181, London, UK, 2002.
5. Jennifer Horkoff and Eric Yu. Finding solutions in goal models: an interactive backward reasoning approach. In *Proceedings of the 29th International Conference on Conceptual modeling (ER'10)*, ER'10, pages 59–75, Vancouver, Canada, 2010.
6. Jennifer Horkoff and Eric Yu. Comparison and evaluation of goal-oriented satisfaction analysis techniques. *Requirements Engineering Journal (REJ)*, 2012.
7. Emmanuel Letier and Axel van Lamsweerde. Reasoning about partial goal satisfaction for requirements and design engineering. In *Proceedings of the 12th International Symposium on the Foundation of Software Engineering FSE-04*, pages 53–62, Newport Beach, CA, 2004.
8. Sotirios Liaskos, Rina Jalman, and Jorge Aranda. On eliciting preference and contribution measures in goal models. In *Proceedings of the 20th International Requirements Engineering Conference (RE'12)*, pages 221–230, Chicago, IL, 2012.
9. N.A.M. Maiden, P. Pavan, A. Gizikis, O. Clause, H. Kim, and X. Zhu. Making decisions with requirements: Integrating i* goal modelling and the AHP. In *Proceedings of the 8th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ'02)*, Essen, Germany, 2002.
10. John Mylopoulos, Lawrence Chung, and Brian Nixon. Representing and using nonfunctional requirements: A process-oriented approach. *IEEE Transactions on Software Engineering*, 18(6):483–497, 1992.
11. Eric S. K. Yu. Towards modelling and reasoning support for early-phase requirements engineering. In *Proceedings of the 3rd IEEE International Symposium on Requirements Engineering (RE'97)*, pages 226–235, Annapolis, MD, 1997.

A Systematic Comparison of *i** Modelling Tools Based on Syntactic and Well-formedness Rules

Catarina Almeida, Miguel Goulão, and João Araújo

CITI, FCT, Universidade Nova de Lisboa, Portugal
acg.almeida@campus.fct.unl.pt
{mgoul,joao.araujo}@fct.unl.pt

Abstract. There are several tools currently available in the *i** community. These tools have different features and purposes. Choosing the most adequate tool for a specific modelling situation can be a challenge. To overcome this difficulty, we present a systematic comparison of the *i** tools listed in the *i** wiki page, according to their features, syntax coverage and semantic analysis support. Our comparison highlights the different strengths of those tools, to help identifying situations for which each tool might be particularly useful. We contribute with an aggregated vision of current *i** tool support to the body of knowledge of the *i** community. In addition, this comparison also helps identifying opportunities for further evolution of the surveyed tools.

Keywords: *i** modelling framework, systematic comparison, tool support

1 Introduction

The *i** framework is a modelling language that covers both agent-oriented and goal-oriented modelling. There are several variations of the *i** framework, such as *Yu'95*, *TROPOS*, *Secure Tropos*, *Iterative Tropos*, and *GRL*. There are also several tools currently available to create *i** models. Those tools have different features, purposes, and various levels of conformity with the *i** syntax (usually alligned with one of the above-mentioned *i** frameworks).

Choosing the best tool for a specific purpose can be a challenge, not only because one has to select the most suitable *i** framework for the task, but also because different tools targeted to the same *i** framework provide different kinds of support for the specification of an *i** model, not only on the syntactic, but also on the semantic level. This support includes different levels of correctness checking of the created models. The *i** wiki page [1] includes a comparison of the *i** tools to address this challenge, covering information such as the purpose of each tool, the *i** framework it supports, practical details concerning availability, base platform, maturity, etc., as well as details on the tool modelling suitability, usability, extensibility and interoperability.

In this paper we present a systematic comparison of syntactic and semantic features supported by the different *i** tools. We use the language description

available from the i^* wiki to guide our comparison. This provides a language-oriented comparison of the i^* tools, which bridges an important gap in the scope of the comparison currently available in the i^* wiki.

This paper is organized as follows: in section 2, we summarize the objectives of this i^* tools comparison; in section 3 we provide a detailed comparison of the i^* tools, focusing on the syntactic coverage and semantics checking of the i^* models; finally, in section 4, we present conclusions and further work.

2 Objectives of the Research

Our goal is to analyze i^* modelling tools, for the purpose of their comparison, with respect to their syntax coverage, and semantic checking support from the point of view of a requirements engineer, in the context of a survey of the existing tool support available to the i^* community.

More specifically, we aim to answer the following two research questions:

- **RQ1:** Which of the syntactic constructs described in the i^* wiki are supported by the i^* tool?
- **RQ2:** To what extent does each i^* tool support semantic checking of the i^* models built using it?

3 Scientific Contributions

In order to answer our research questions, we tested available i^* tools. The criteria used for the inclusion of the tools in this analysis was their presence in the i^* wiki page [1] and the availability of a functional URL. Some of the tools refered in the i^* wiki are no longer available in the advertised URL, and were therefore excluded from this analysis. Table 1 shows the different tools surveyed in this paper.

Table 1. Analysed Tools

i* **Tool**	**Institution**	***i**** **Variant**	**Platform**			**Technology**
			Windows	**Linux**	**MacOS**	
OpenOME	Univ. Toronto	Yu'95	Yes	Yes	Yes	Java (JRE)
TAOM4E	Univ. Trento	Tropos	Yes	Yes	Yes	Eclipse plug-in
GR-Tool	Univ. Trento	Tropos	Yes	Yes	Yes	Java (JRE)
STS-Tool	Univ. Trento	Tropos	Yes	Yes	Yes	Java (JRE)
jUCMNav	Univ. Ottawa	GRL	Yes	Yes	Yes	Eclipse plug-in
DesCARTES	U.C. Louvain	Yu'95 / Tropos	Yes	Yes	Yes	Eclipse plug-in

OpenOME [2] is an Eclipse-based tool designed to support goal-oriented, agent-oriented and aspects-oriented modelling and analysis. It is an open source tool and researchers can propose further branches and extensions to the tool. TAOM4E [3] is an Eclipse plug-in that supports a model-driven, agent oriented software development. GR-Tool [4] is a graphical tool for forward and backward goal reasoning in Tropos. STS-Tool [5] is a socio-technical security modelling tool to draw Tropos and Secure Tropos models and to perform the effective formal analysis of functional and security requirements. jUCMNav [6] is an Eclipse plug-in for modelling, analysis and transformation in both GRL and UCM (Use Case Map) notation. It is intended for the elicitation, analysis, specification and validation of requirements. DesCARTES [7] is an Eclipse plug-in that supports various models, such as *i**, NFR, UML and AUML models in the context of Tropos and I-Tropos development. The tool allows the development of the methodology analysis and design models as well as forward engineering capabilities and an integrated software project management module.

All the tools were tested in Arch Linux x86_64 and Windows Vista 32 bits, both with Java Runtime Environment 1.6.0. For the tools that are an Eclipse plug-in, the version of the Eclipse was Indigo (3.7) for TAOM4E and Juno (4.2) for jUCMNav and DesCARTES, with Eclipse Modelling Tools.

The syntax coverage analysis aims to check if the tool has *a*) the basic *i** syntax; and *b*) the graphical notation of the *i** syntax (according to the *i** wiki page). Table 2 shows this analysis. The cells with "Yes" denote that the tool complies with both criteria presented earlier. The cells with the symbol "*" after "Yes" mean that the tool complies with *a)* but not with *b)*; i.e., it has a different notation for the given element. The cells with "No" show that the tool does not comply with any of the criteria.

The GR-Tool is the only not supporting any type of actor, as it is mostly targeted to goal reasoning. OpenOME is the only tool that supports all types of actors with the *i** graphical notation. Actors links are often not provided and are only fully supported in OpenOME and jUCMNav. The STS-Tool also supports a type of actor association, namely the association "plays".

Concerning elements, all the tools support goals. Only jUCMNav has all kinds of elements of the *i** graphical syntax. DesCARTES also supports all element types, but two of them use a non-standard notation.

With respect to the support for dependency links, the tools which support them only have commited elements, but not open or critical elements. The GR-Tool and STS-Tool do not support any kind of dependency links.

All the tools have at least two types of contribution links and all of them have the "and" link. It is in the contribution links that the variation of the notation according to the graphical notation of *i** syntax is higher. OpenOME and jUCMNav are the only tools that have all types of contribution links.

As can be observed, OpenOME is the tool with the widest syntax coverage according to the two criteria described above and if it complies with the first one, it always complies with the second. jUCMNav complies with the first criteria except for dependency strengths but not always complies with the second one.

Table 2. *i** Syntax Coverage

	OpenOME	TAOM4E	GR-Tool	STS-Tool	jUCMNav	DesCARTES
Actors						
Actor	Yes	Yes	No	No	Yes	No
Actor Boundary	Yes	Yes	No	No	Yes	No
Role	Yes	No	No	Yes	Yes*	No
Agent	Yes	No	No	Yes	Yes*	Yes
Position	Yes	No	No	No	Yes*	No
Actors Links						
ISA	Yes	No	No	No	Yes*	No
Is-part-of	Yes	No	No	No	Yes*	No
Plays	Yes	No	No	Yes	Yes*	No
Covers	Yes	No	No	No	Yes*	No
Occupies	Yes	No	No	No	Yes*	No
INS	Yes	No	No	No	Yes*	No
Elements						
Goal	Yes	Yes	Yes	Yes	Yes	Yes
Softgoal	Yes	Yes	No	No	Yes	Yes*
Task	Yes	Yes*	No	No	Yes	Yes
Resource	Yes	Yes	No	No	Yes	Yes
Belief	No	No	No	No	Yes	Yes*
Links						
Dependency Links	Yes	Yes	No	No	Yes	Yes
Means-ends Links	Yes	Yes	No	No	Yes*	Yes
Decomposition Links	Yes	Yes	No	No	Yes	Yes
Contribution Links						
Make	Yes	No	No	No	Yes	Yes
Break	Yes	No	No	No	Yes	Yes
Unknown	Yes	No	No	No	Yes	Yes
Some+	Yes	No	Yes*	No	Yes	Yes*
Some-	Yes	No	Yes*	No	Yes	Yes*
Help	Yes	No	Yes*	Yes*	Yes	Yes*
Hurt	Yes	No	Yes*	Yes*	Yes	Yes*
And	Yes	Yes	Yes	Yes	Yes	Yes
Or	Yes	Yes	Yes	Yes	Yes*	No
Dependency Strengths						
Open Element	No	No	No	No	No	No
Commited Element	Yes	Yes	No	No	Yes	Yes
Critical Element	No	No	No	No	No	No

The semantic analysis aims to determine the level of corretness checking of the created models. Using the descriptions and guidelines available in the *i** wiki, we analysed if the tool verifies when a modelling error is made. Table 3 shows a set of errors and if the tool verifies those errors or not. The N/A means that the tool does not have one or more elements needed to do the evaluation.

Table 3. Semantic Analysis

	OpenOME	TAOM4E	GR-Tool	STS-Tool	jUCMNav	DesCARTES
Actors and relations						
Actors without links	Yes*	No	N/A	No	No	No
Actor inside another actor boundary	Yes	Yes	N/A	Yes	No	Yes
Dependencies						
Dependency link without a dependum	Yes*	Yes	N/A	N/A	Yes	Yes
Dependency links with different directions	No	Yes	N/A	N/A	No	Yes
Dependency link inside an actor boundary	Yes*	Yes	N/A	N/A	Yes	N/A
Other link rather than dependency link between an element and an actor	Yes	Yes	N/A	Yes	No	Yes
Associations						
ISA between actors of different types	No	N/A	N/A	N/A	Yes	N/A
Is-part-of between actors of different types	No	N/A	N/A	N/A	Yes	N/A
Other association rather than Plays between Agent and Role	No	N/A	N/A	Yes	Yes	N/A
Other association rather than Covers between Position and Role	No	N/A	N/A	N/A	Yes	N/A
Other association rather than Occupies between Agent and Position	No	N/A	N/A	N/A	Yes	N/A
INS between others than agents	No	N/A	N/A	N/A	Yes	N/A
Associations between elements that are not actors	Yes	N/A	N/A	Yes	Yes	N/A
Internal Elements						
SR elements outside actor boundary	No	Yes	N/A	Yes	No	N/A
Softgoal decomposition in sub-softgoals or sub-tasks	No	No	N/A	N/A	No	Yes
Goal decomposition in sub-goals or sub-taks	Yes*	No	N/A	N/A	No	Yes
Goal decomposition without means-end	No	No	N/A	N/A	No	Yes
Means-end where a goal is the mean	No	No	N/A	N/A	Yes	No
Means-end different from “task–>goal”	No	No	N/A	N/A	No	No
Decomposition beyond the actor boundary	No	Yes	N/A	N/A	No	N/A
Means-end beyond the actor boundary	No	Yes	N/A	N/A	No	N/A
Means-end decomposition to refine a softgoal	No	No	N/A	N/A	No	Yes
Softgoal decomposition without contribution links	No	No	N/A	N/A	No	No
Any kind of direct relation between goals	Yes*	No	N/A	N/A	No	Yes
Link between an element inside the actor boundary and that actor	Yes	Yes	N/A	Yes	Yes	N/A
Contribution Links						
Contribution links between any element to any element rather than softgoal	No	No	N/A	N/A	Yes	No
Contribution link between actors	Yes	Yes	N/A	Yes	Yes	Yes
Contribution link between goals and subgoals or sub-tasks	No	No	No	Yes	Yes	No

Some of the OpenOME categories are labeled with "Yes*". This means that the tool itself does not perform the corresponding analysis by default, but can do it with the add-on "Syntax check". However, syntax checking does not work for Linux and may not work for Mac as the executable version of prolog is not available for these platforms. In jUCMNav, it is necessary that the static semantics checking properties are selected.

*i** tools support error prevention in two different ways. One is by not allowing the user to do what he intends, if it is not correct. The other one is by allowing the user to do so, but inform him that the resulting model is incorrect. In general, the level of correctness checking of the created models is not too deep. Note that the set of modelling errors that can be made depends on the available modelling elements, which vary from one tool to the next. On average, about 39% of the considered modelling errors are not applicable, due to the lack of support of the corresponding modelling elements by the tools.

There are some errors that all the tools (except those that do not support the used model elements) verify. This is the case of a dependency link without a dependum, associations between elements that are not actors, a link between an element inside the actor boundary and that actor, and a contribution link between actors. The opposite can also be noticed, i.e., there are some errors that none of the tools verify. This is the case of means-ends relations different from "task–>goal", softgoals decomposition without contribution links and contribution links between any element to any element rather than a softgoal.

jUCMNav is the tool with the highest number of verified errors, with a verification percentage of 50%, followed by OpenOME and TAOM4E. They are also the tools with the lowest number of errors that were not possible to analyse.

4 Conclusions and Future Work

This paper provides a systematic comparison of some *i** tools, listed in the *i** wiki. The goal here is to complement existing comparison work by providing a comparison on syntatic and semantic properties of the *i** modelling tools.

The contribution of this work is to provide relevant information to practitioners when selecting an *i** tool, to tool developers when improving existing tools, and to researchers when studying the shortcomings of existing tools.

We plan to extend this analysis to other tools and include non-functional aspects in the evaluation.

References

1. *i** wiki: `http://istarwiki.org/`
2. OpenOME: `https://se.cs.toronto.edu/trac/ome/`
3. TAOM4E: `http://selab.fbk.eu/taom/`
4. GR-Tool: `http://troposproject.org/tools/grtool/`
5. STS-Tool: `http://www.sts-tool.eu/`
6. jUCMNav: `http://jucmnav.softwareengineering.ca/ucm/bin/view/ProjetSEG/`
7. DesCARTES: `http://www.isys.ucl.ac.be/descartes/`

Analysing User Feedback and Finding Experts: Can Goal-Orientation Help?

Matthieu Vergne, Itzel Morales-Ramirez, Mirko Morandini, Angelo Susi, and Anna Perini

Fondazione Bruno Kessler
Software Engineering Research Unit
Via Sommarive 18, 38123 Trento, Italy

Abstract. Goal-oriented approaches in requirements engineering aim at understanding stakeholders' needs, modelling the intentions, dependencies and expectation to be met by a system-to-be, or by a new release of an existing system in a software evolution process. In the context of software evolution, we consider user feedback, as commonly available in user forums and bug-trackers, as the information artefact that impacts specific requirements and design of the software. In this position paper, we argue about the advantages that goal-orientation can bring in this context when addressing the following issues: i) the analysis of user feedback, usually expressed as free or semi-structured text; and ii) the identification of the most expert users that can contribute to requirements evolution. We define the problems, state the emerging research questions and present contributions with the help of an illustrative example.

1 Introduction

Goal-oriented (GO) approaches in requirements engineering (RE) aim at understanding stakeholders' needs, by modelling the intentions, dependencies and expectations to be met by a system-to-be or by a new release of an existing system. They are widely adopted in requirements elicitation activities based on face-to-face meetings with stakeholders (e.g. users and analysts), e.g. in [1], supporting the communication while acquiring and refining requirements. After releasing a software application, the activities of acquiring and analysing requirements for the purpose of system maintenance and evolution remain still important, especially as long as users provide their feedback to the analysts team, which is the case in many open-source projects. In the context of our research, we refer to user feedback as an information artefact communicated as free or semi-structured text, that expresses the users' perspective upon experiencing the use of a software application. This feedback is usually analysed against the system's software architecture and code for the purpose of bug fixing and general system maintenance. Yet, feedback can also be used for deriving the requirements of next versions [2].

Currently, as far as we know, there is no link between this feedback and a requirements model that supports a classification of feedback and the organisation of it. We are investigating on user feedback from a RE perspective, with the aim of defining a structured approach for managing it. We believe that requirements models can help analysts

to structure user feedback and to identify the most expert users (among the forum's participants) who could be engaged for clarifying or evaluating new requirements.

The main issues to be faced towards this objective are the separation of relevant and noisy feedback and the definition of criteria for ranking forum participants against expertise levels. We identified various specific challenges in analysing feedback, including: i) coping with the heterogeneous abstraction levels in which it can be expressed; ii) identifying the key concepts in feedback that can possibly be linked to an existing requirements specification; and iii) evaluating its impact on system requirements [3]. Concerning expert finding, challenges are: iv) identifying relevant pieces of knowledge and their relationships for the purpose of modelling the expertises of the users; and v) building a consistent model that we can query to infer the most expert people in the group under consideration.

In this paper, we focus on the challenges *ii)* of feedback analysis and *v)* of expert finding, discussing about advantages given by adopting a GO approach. We make the strong assumption that, for a given software application, a GO requirements specification is kept aligned along the phases of system maintenance and evolution.

We envisage a process in which the feedback is analysed and correlated to the key concepts in a project's requirements model. In a second step, the links obtained from this analysis can be exploited, among other data, to identify expert users that can contribute effectively to the requirements analysis.

To approach the first challenge we propose a meta-model that defines the underlying structure of user feedback, and relate it to the conceptual entities of the GO modelling language Tropos [4], while for the second challenge we exploit Markov networks for processing the information available in forums and goal models in a probabilistic way, which will support making inferences about users' expertise.

In Section 2, we state the research objectives and describe an illustrating scenario. Section 3 presents the contributions along the two lines of feedback analysis and expert finding, while Section 4 concludes and outlines future work.

2 Research objectives and motivating example

2.1 Objectives

We consider the analysis of user feedback may be guided by a meta-model, thus identifying its purpose and impact on a goal model. On the other hand, the combination of this feedback with the goal model may lead to identify the potential experts on relevant topics, supporting a better requirements refinement process. Consequently, we want to discuss:
– how we can benefit from a *user feedback meta-model*, centred around the concept of *purpose* and a set of *bridging links* to the Tropos meta-model, to understand to what extent the user feedback impacts on a requirements specification;
– how probabilistic models such as *Markov networks* can be used to exploit the information provided in a GO approach in order to identify expert users participating in forum discussions, and on which we can rely for the improvement of the system.

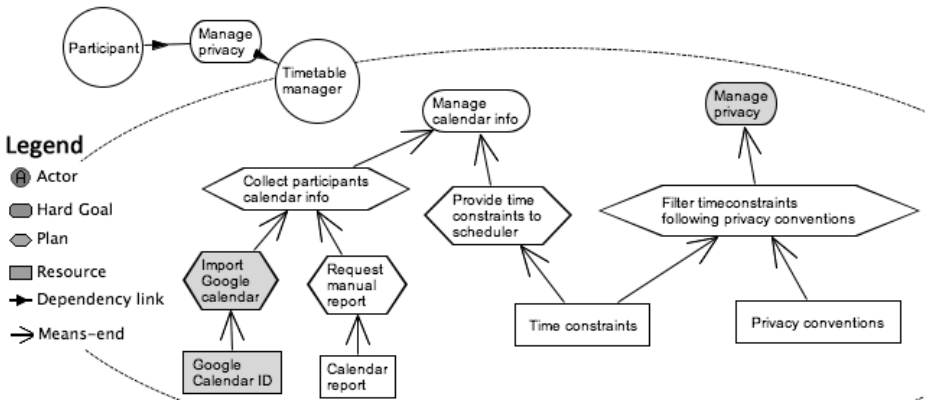


Fig. 1. Internal goals of the *Timetable manager* actor (Tropos notation).

2.2 Motivating example

We envision a software application, called OPEN-MEET, that schedules internal meetings in research groups. This application has been developed adopting the Tropos GO approach for domain analysis and requirements modelling. An excerpt of the model is depicted in Figure 1, showing the functionalities performed by the actor "Timetable manager" (a sub-component of OPEN-MEET). The system schedules meetings by considering the personal calendar of all the people in the group (the participants), as well as the availability of meeting resources. To improve the app, its users can provide feedback in a forum (bug reports, suggestions, wishes, etc.). To do so, they write their concerns, in such a forum, by giving a short title and a description. Examples of user feedback are:
-Stefania "It would be nice to have an option to specify certain days of the week on which I'm fully available" (Concerning calendar information that users should be able to provide).
-Paolo "I do not want my full Google calendar to be considered, only the periods related to my working time" (Concerning privacy issues).

3 Scientific contributions

3.1 Analysis of User Feedback

We want to analyse the user feedback and understand whether it impacts on requirements expressed in a goal model. In a previous work [3] we reviewed the literature on feedback and derived a conceptualisation to describe feedback's rationale and structure. We extend this conceptualisation in terms of a meta-model with *bridging links* to the Tropos meta-model, as described in [4]. Excerpts of both meta-models are shown in Figure 2, left side depicts the conceptual entities that drive the analysis of feedback, right side shows Tropos excerpt. User feedback is elaborated in natural language and

typically based on some *topics*[1]. From our perspective, user feedback can have one or more *purposes*. We present only three purposes: (1)Improvement, (2)Clarification and (3)StrategicBehaviour. Number 1 refers to users' wish of using a better app, 2 for requesting further details (both ways), and 3 refers to the propagation of the impact through system's functional and non-functional requirements that can receive an action, for instance modification. The *bridging links* are high-level expressions of concepts contained in feedback. For example, FunctionImprovement may refer to a specific behaviour of the app, e.g. "print calendar". QualityImprovement may refer to a judgement about the operation of the app, e.g. "calendar interface". ExecutionImprovement may refer to the steps for performing a function, e.g. "first select day". Therefore, we see that the *bridging links* are linked to a HardGoal, SoftGoal, Plan, respectively. Finally, in the feedback excerpt "It would be nice... an option", it may refer to a new functionality because of the expression highlights a wish, i.e. a *goal*. The entity OwnedBehaviour is connected to the entity Decomposition to indicate that an Action might be applied to an element of Tropos by following the means-end links, i.e. to propagate a likely impact.

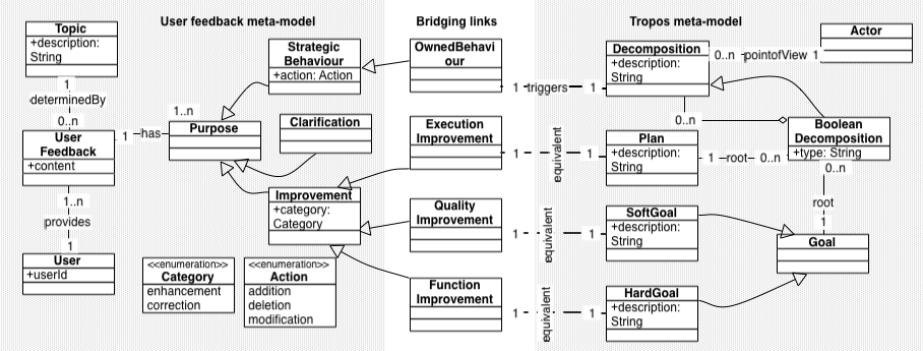


Fig. 2. Excerpt of the user feedback meta-model with bridging links to the Tropos meta-model.

Recalling the running scenario, Paolo's feedback could be classified under the *topic* "Manage privacy", which turns out to be a *goal* (see Figure 1). The meta-model may help in guiding the identification of the *purpose(s)* contained in such a feedback, thus leading to the impacted fragment in the model. In our example, the purpose is a written expression recognised by the starting words "I don't want". This expresses a wish of avoiding something. The key concepts "full Google calendar" and "to be considered" point out that there is a request to avoid considering the whole calendar. As Paolo's concern may be a correction, the entity OwnedBehaviour triggers first the propagation with the link modification to a functionality in the model that contains the text *calendar*, i.e. the goal "Manage calendar info". By following the means-end links, the analyst can deduce that the feedback "...full Google calendar..." may refer to modify a *plan*. So the impacted fragment to be analysed is the plan related to the goal "Manage calendar info". Since the feedback was provided under the topic "Manage privacy", it needs to be

[1] The elaboration of the *topic* based on system's behaviour is an issue under research.

classified as Clarification and discuss it with the users to validate requirements change. But at this point the analyst will be able to identify likely impacted fragments (elements shown in Figure 1[2] in grey).

3.2 GO-based Expert Finding

By investigating expert finding techniques in RE[3], we have identified several types of relationships to exploit, as for instance the concepts related to a broad topic or the topics a user knows about. In this section, we show that the goal model and the analysis of the feedback can provide such relationships, although there is uncertainty issues, in order to identify *expert users* for the discussion process. Similarly to [5], we build a *Markov network*, able to exploit these relations and manage uncertainty, with the information provided by the goal model and the previous feedback (in the forum), relating the users to different *pieces of knowledge* (e.g. concepts and topics). The analysis of the current user feedback is then used to identify the pieces of knowledge to consider, to query the network to infer the probability that a given user suits to the discussion, and finally to rank the users regarding these probabilities.

Concretely, we extract pieces of knowledge from the goal model, which are *concepts* (in the labels of the goals, plans and so on), *topics* (concepts in not-leaf elements like the goal "Manage privacy") and abstractions of the users (i.e. actors) that we will call *roles*. We can also infer their *relationships* through this goal model: for instance, as a goal decomposition relates a parent goal to its sub-goals, it also relates the topics of the former to the concepts of the latter. Then, we exploit the previous feedback contained in the forum, identifying further relations between topics and concepts, but also the *users* who have been involved and their relations to the corresponding topics and concepts (we can use other sources to identify user-to-role relations).

Consequently, we have several relevant pieces of knowledge, namely users, roles, topics and concepts, and several ways to relate them: each user can play some roles, know about some topics and use some concepts; playing a specific role can lead to know about some topics and concepts; knowing about a specific topic can lead to know some concepts. Thus, considering that each one corresponds to a *node* in a graph, we can then relate all the nodes of one kind to all the nodes of another kind, making the graph almost fully connected (only nodes of the same kind are not related, e.g. a user is not related to another user). The amount of information we have for each relation can represents its *strength*, like in a weighted graph, and more this strength is high more the data is reliable. For example, if a lot of feedback on the topic "Manage privacy" mentions "Google calendar" but a few mentions "lunch", the link between the former will be stronger than for the latter. Finally, the strong paths linking the users to the different pieces of knowledge indicate which users are better to involve, considering a set of topics or concepts we are looking for.

We model this kind of graph via Markov networks, where the nodes are *random variables* and the relations correspond to *potential functions*. In our case, the random variables are binary, identifying whether a user/role/topic/concept is concerned

[2] This is a late requirements model that represents the results on the elaboration of feedback.

[3] *Infer Informational Capabilities by Relating Expertises in Requirements Engineering*. Technical report. Matthieu Vergne, 2013. Document available on request.

by the discussion process or not, and the potential functions compute the strength of the edges depending on the state of the related nodes. With such a model, we can infer the *probability* (which is the normalised product of the potential functions) for a specific user to be of interest given some identified pieces of knowledge, formalised as $P(user|roles, topics, concepts)$. Computing this probability for all the users, we can rank them to identify who are the best ones to recommend for the discussion. For instance, if the analyst identifies with the feedback the topic "Manage privacy" and the concept "Google calendar", he computes $P(x_i|\text{Manage privacy}, \text{Google calendar})$, where x_i corresponds to Stefania, Paolo or any other user, and finds the most suitable users by looking at the highest probabilities.

4 Conclusion and future work

In this paper, we provided a preliminary discussion about the advantages of linking a meta-model of user feedback with the conceptual entities of the Tropos meta-model. We also discussed how a GO approach can be exploited to model the relevant knowledge in a Markov network, and how it relates to the available users to infer the most probable experts.

So far, the proposed meta-model supports a more focused analysis of feedback, by revealing a hidden structure that can be exploited to identify the fragments of a requirements model which are affected by the feedback. The expert finding, on the other hand, supports the understanding of how this model should be impacted, using the feedback analysis to identify expert users to discuss with. However, the analysis is performed manually and the use of the meta-model is merely focused on giving a structure to user feedback, while the Markov network still need to be parametrised and assessed with some quality measures.

As future work, we plan to support a semi-automatic identification of feedback purpose, by classifying it with the help of feedback patterns combined with a supervised learning process, and try to identify parameters that provide a good compromise between ranking quality and computation time.

References

1. Morales-Ramirez, I., Vergne, M., Morandini, M., Sabatucci, L., Perini, A., Susi, A.: Where did the requirements come from? a retrospective case study. In: ER Workshops. Volume 7518 of LNCS., Springer (2012) 185–194
2. Schneider, K.: Focusing spontaneous feedback to support system evolution. In: RE, IEEE (2011) 165–174
3. Morales-Ramirez, I.: On exploiting end-user feedback in requirements engineering. Doctoral Symposium at REFSQ13. ISSN 1860-2770. (April 2013)
4. Giorgini, P., Mylopoulos, J., Perini, A., Susi, A.: The Tropos Methodology and Software Development Environment. In Yu, Giorgini, Maiden, Mylopoulos, eds.: Social Modeling for Requirements Engineering, MIT Press (2010) 405–423
5. Hu, D.H., Yang, Q.: Cigar: Concurrent and interleaving goal and activity recognition. In Fox, D., Gomes, C.P., eds.: AAAI, AAAI Press (2008) 1363–1368

Using i* Models to Enrich User Stories

Aline Jaqueira[1], Márcia Lucena[1], Eduardo Aranha[1],
Fernanda Alencar[2] and Jaelson Castro[3]

[1]Departamento de Informática e Matemática Aplicada – UFRN
alinejaqueira@ppgsc.ufrn.br, {marciaj, eduardo}@dimap.ufrn.br
[2]Departamento de Eletrônica e Sistemas - UFPE
fernanda.ralencar@ufpe.br
[3]Centro de Informática - UFPE
jbc@cin.ufpe.br

Abstract. In agile methods the user stories are widely used to describe requirements. However, the user stories are an artifact too narrow to represent and detail the requirements. Issues like software context and dependencies between stories are also limited with the use of only this artifact. The lack of documentation in agile development environment is identified as one of the main challenges of the methodology. This work proposes the use of i* model that aims to reduce this lack of existing documentation in agile methods. We propose a set of heuristics to perform the mapping of the requirements presented as user stories in i* models. The i* models are used as a form of documentation in agile environment, thus the user stories can be viewed more broadly and with their proper relationships according to the business environment that they will meet.

Keywords: i* Models, User Stories, Agile Requirements

1 Introduction

Agile methods have been gaining much interest among practitioners and researchers [1], because they use more simplified processes with less bureaucratic activities associated with the development [2]. However, requirements engineering and agile development are often seen as incompatible activities [3], as requirements engineering is a traditional process of software engineering and has the documentation to manage the project and share knowledge, while the agile methods focus on face-to-face collaboration among stakeholders to address the same goals.

In addition, the elicitation is held with clients that are part of the development team. In order to do this, customers write stories according to the system needs to do (user stories) and prioritize them according to the value of the concerned business. A user story describes the functionality that is valuable to the customer and is used for project planning, acting as a reminder to the team since subsequent conversations about the story are essential to convey the details [4]. User stories are used to establish a common understanding of the software requirements using a flexible approach,

low overhead and focusing on the user [5]. They are widely used by agile development [6] and are therefore considered in this work as an agile artifact.

The requirements documentation is seen as a bureaucratic activity in the Agile methods [2], but their absence is singled out as one of the main challenges of the methodology [8]. In addition, the user stories are very limited artifacts to process issues such as progress tracking software and that there is a lack of detailed information about software in development [9]. The control of the system performed only by way of user stories is a challenge both for customers and for the team. To make decisions based only on the stories, without any documentation, becomes risky especially for complex systems [14].

This work proposes the use of i* model that aims to reduce this lack of existing documentation in agile methods. We propose a set of heuristics to perform the mapping of the requirements presented as user stories in i* models. The i* models are used as a form of documentation in agile environment, thus the user stories can be viewed more broadly and with their proper relationships according to the business environment that they will meet.

In the following section, briefly define the research objectives. In Section 3 we discuss the scientific contributions. Section 4 provides the conclusions and Section 5 presents future work.

2 Objectives of the research

This work proposes the use of i* models as an additional resource that aims to reduce the lack of documentation of agile software development projects and improve view dependencies between user stories and the system and also provide information and understanding of the system as a whole. We seek to support the stakeholders, through models i* providing a graphical and comprehensive vision of the user stories and their relationships.

The common concern with stakeholders justifies the choice of the technique i* to represent the user stories. The focus on stakeholders and their relationships to the description of requirements is a feature of the technique i*, where actors depend on each other to achieve their goals. The agile methods also focus on human factors and bet in delivering value to the customer [10], recognizing people as fundamental part of the project success [11]. Therefore, the concepts of user stories are employed together with the concepts of the i* model [12].

Although user stories are written in natural language by the client, Mike Cohn suggests a format for writing the stories that has been widely used: "as <role>I want <action>to <goal> " [7]. Therefore, for this proposal it is assumed that the same format proposed by Cohn [7] will be used.

The user stories are mapped to i* models generating a graphical and comprehensive vision of the software requirements and their relationships. The concepts and notations of the i* model are used according to i* Wiki [13], that represents a simplified version of the technique. It is also important to report that only a few elements of i* are used in accordance with the need of the proposal. In this way, to build the SD

model, the elements used are the actor, the goals and the IS_A Association. For SR model the tasks, resources, and also the connection of decomposition are used.

By using i * models from user stories, agility feature of the projects should be maintained because the view of stories complemented by i * models can make better understanding, easier and simpler. The following presents the correspondence of elements when mapping the user stories to i* model: (i) Role in User Story is mapping to Actor in i* model; (ii) Action in User Story is mapping to Task in i* model; and (iii) Goal in User Story is mapping to Goal in i* model.

To simplify the understanding of the mapping, heuristics were established as a resource to arrive at the solution of the mapping proposed in this paper. It is noteworthy that the heuristics set must be executed in the order they were made for the mapping occur in a more objective. The heuristics to map user stories to SD model are: SD-H1: Create the Actor System; SD-H2: Create an Actor in i * model for each different role of user stories; SD-H3: Create a meta in i* model for each goal of user stories. If repeated the same goals will be defined only once in the model; SD-H4: If repeated goals for different actors, create a Generic Actor; SD-H4.1: Create a relationship is_a the Actor generic to other specific actors who share the same goal; SD-H5: Connect the dependencies of each actor with their goals.The heuristics used to map user stories for the SR model are: SR-H1: Create a Task within the Actor System for each share of user stories; SR-H2: If there are different actions for the same goal, to create a generic task; SR-H2.1: Decomposing the generic task into sub tasks that represent the actions associated with the same goal; SR-H3: List the dependencies of each goal with the corresponding tasks according to user stories; SR-H4: If there are tasks that depend on own Actor that are related, generate a resource with the name of the task; SR-H5: Relate the resource depending on the Actor.

According to the proposed heuristics, the SD model is mapped by first creating the System actor. Subsequently, it creates the actors for the roles of user stories, which, in this case, the actors are User and Administrator. The goals of the user stories are created as goals for the SD model and linked as dependencies leaving from the actors that are associated and coming to System Actor. We omitted the result of heuristics this example for SD model mapping (from user stories to i* SD model) due to lack of space.

To demonstrate this proposal a login system was used as an example, considering the prospect of a user and an administrator. Table 1 presents the stories of user login system and the figure 1 shows the result of the mapping.

Table 1. User stories Login System (Source: IBM, 2012)

	Role	**Action**	**Goal**
1	User	Having username	Access secure content
2	User	Having password	Access secure content
3	User	Choose your username	Customize account
4	User	Change the default password	Customize password
5	Administrator	Assign the user password	Automated registration
6	Administrator	Send email registry	Confirm the account activation email
7	Administrator	Request to login user	Ensuring security of con-

			tent
8	User	Register password reminder	Remember the password
9	Administrator	Request password reminder	Confirm user

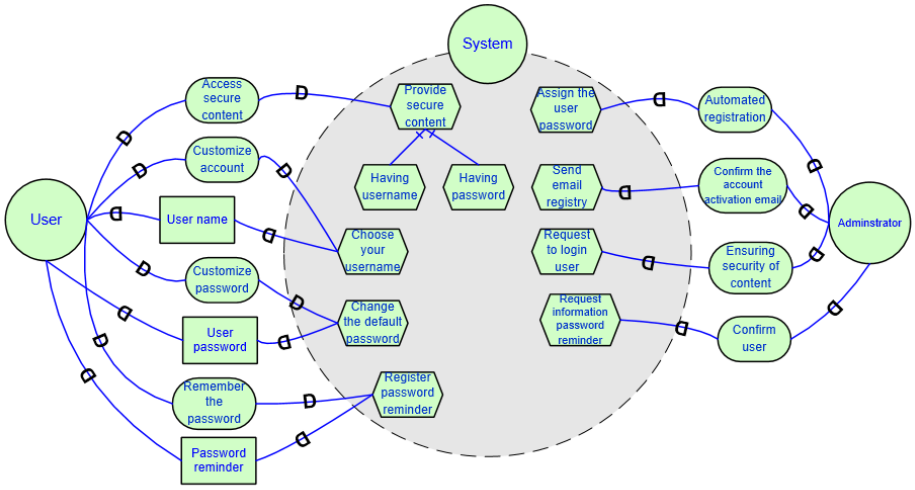


Fig. 1. Model SR in the sample application

To generate the SR model, every action of the user stories have been generated as a task within the system actor, once it is the system actor that will operate them, performing the task in a particular manner in order to meet the goals of the actors. As there are, in this example, different actions for the same goal, a generic task was created in SR model which was decomposed in the actions in the form of sub-tasks. The tasks that depend on the actor himself generate a resource that depends on the actor and that has the same name as the task.

3 Scientific contributions

Even in an agile environment it is necessary to develop some models before any implementation to ensure a shared understanding by the development team, so that it is synchronized with the goals of the business value and context of the project [14]. Visual models assist in understanding of how users will need to use the system. In addition, those models are effective for the stakeholders to understand the proposed solution and also to keep them interested and involved.

The most important contribution of this work is the development of a proposal that uses visual models provided by i* to alleviate the lack of documentation in agile development environments that was cited in the systematic review conducted by Jaqueira et al. [8]. A set of heuristics is proposed to perform the mapping of user stories for i* models.

Since i * models are graphical representations of the requirements, they would have a comprehensive and visual way to see the user stories, thus mitigating the lack

of documentation in agile software development projects, improving the visualization of the system context, facilitating access to requirements, contributing to decision-making in the development environment.

In addition, other contributions may be cited as the improvement in understanding the context of the system to be developed, the use and easier access to information of user stories from the preview of the visual model, improving the decision-making process in accordance with the analysis of the stories, as they are described in i* models, and tooling support enabling the automation of some of its stages (construction of SD and SR models).

4 Conclusions

To evaluate the approach this work, we performed a case study analyzing qualitative issues. From the participants' impressions of use, it was found that the use of i * models contribute to complement user stories [16]. An approach based on visual models can provide more direct and traceable links to system development, promoting an analysis with the greatest impact in the design and implementation of software. It also works to facilitate communication, understanding, and detecting problems or explore what-if scenarios and potential solutions.

We find that when viewing models, errors and/or neglect could be more easily recognized [15]. All this facilitates the process of analysis and discussion of the system to be developed.

From the mapping of the user stories to SD and SR i* models, we can organize and represent all the stories in a model that provides an overview of the stories and their relationships. In addition, all the stories of the same actor were presented in the same model, allowing to find them more easily. In this way, it is possible to understand the context of the system, its main actors and their goals.

Viewing through the models makes it easier to identify dependencies between user stories and the identification of system tasks to meet each specific actor involved with the software. Thus, it is possible to notice that the use of i* models enriches the user stories to provide a better view (broader and general); to enable showing the dependencies between the stories, contributing to a better understanding of the context of the system to be developed; to provide visualization of system tasks associated with the goals of each actor; to allow the recognition of possible errors or negligence in the stories. Therefore, this work proposes a form of documentation of the requirements on agile development, thus a visual artifact will be provided supplementing the user stories allowing analysis, communication, discussion and better understanding of the system.

5 Ongoing and future work

In order to continue the research for this work a few suggestions of further work can be cited. The development of a tool to perform the transformation of user stories the format Mike Cohn [7] used in this work, in order to use this proposal to all user sto-

ries. The development of a tool for the purpose of this work in order to fully automate the mapping of user stories for i* models. The treatment of scalability for the system actor. Identify and treat the relationships and connections between tasks in System Actor. Furthermore, the development of guidelines to perform the mapping back from i* models to user stories.

References

1. Cao, L. and Ramesh, B., Requirements Engineering Practices: An Empirical Study, IEEE Software, Volume: 25, Issue: 1. page(s): 60-67 (2008).
2. Fowler, M. The new methodology. Disponível em: <http://www.martinfowler.com/articles/newMethodology.html>. Acesso em 03/12/2011.
3. Paetsch, F., Eberlein, A. and Maurer, F. Requirements Engineering and Agile Software Development, Proc.12th IEEE Int'l Workshops Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 03), IEEE CS Press, 2003, pp. 308–313.
4. Cohn, M.: User Stories Applied: For Agile Software Development (The Addison-Wesley Signature Series), March. Addison-Wesley Professional, Reading (2004).
5. O'hEocha, C., & Conboy, K. (2010). The role of the user story agile practice in innovation. In P. Abrahamsson & N. Oza (Eds.), Lean enterprise software and systems (pp. 20-30). New York: Springer.
6. Cockburn, A. (2007). Agile Software Development: The Cooperative Game. Boston, Pearson.
7. Cohn, M. Agile Estimating and Planning. Prentice Hall, 2006.
8. Jaqueira A., Andreotti, E., Lucena, M., Aranha, E. Desafios de Requisitos em Métodos Ágeis: Uma Revisão Sistemática 3rd WBMA, São Paulo, 2012.
9. Sharp, H., Robinson, H. Segal, J. and Furniss, D. (2006) 'The Role of Story Cards and the Wall in XP teams: a distributed cognition perspective', Proceedings of Agile 2006, IEEE Computer Society Press, pp65-75.
10. Bassi, D. Experiências com desenvolvimento ágil, Dissertação de Mestrado, Universidade de São Paulo, 2008.
11. Highsmith, J. and Cockburn, A., "Agile Software Development: The Business of Innovation," Computer, vol. 34, pp. 120-122, 2001.
12. Yu, E. "Modelling Strategic Relationships for Process Reengineering". PhD thesis.University of Toronto, Department of Computer Science. 1995.
13. i* Wiki Home, Disponível em <http://istar.rwth-aachen.de/tiki-index.php> Acesso em 10/08/2012.
14. Beatty, J. e Chen, A. Visual Models for Software Requirements. Washington, Microsoft Press: 2012.
15. Horkoff, J., Yu, E.: A Qualitative, Interactive Evaluation Procedure for Goal- and Agent-Oriented Models. In: CAiSE Forum. CEUR Workshop Proceedings (2009).
16. Jaqueira, A.: Uso de Modelos i* para Enriquecer Requisitos em Métodos Ágeis. Dissertação de Mestrado. Departamento de Informática e Matemática Aplicada – UFRN. Natal (2013)

On the Integration of i* into RUP

Yves Wautelet[1] and Manuel Kolp[2]

[1] Hogeschool-Universiteit Brussel, Belgium
yves.wautelet@hubrussel.be,
[2] Université catholique de Louvain, Belgium
manuel.kolp@uclouvain.be

Abstract. Altought widely used and recognized in the scientific community, the i* framework has, until now, failed to impose itself into enterprise practices. There are many ways that can be followed to favor industry-adoption. Among them, we believe that an integration into the (Rational) Unified Process, which already includes business modeling as a preliminary step in software development and furnishes custom syntax and semantic to do so could be an interesting approach. This paper summarizes the ideas of a research aimed at mapping i* model elements and RUP/UML business modeling ones with the best possible semantic match. The willingness is to provide RUP practitioners a powerful tool for capturing and analyzing social and organizational contexts of software systems based on the syntax they already know with as closely as possible related semantics.

Keywords: i*, RUP Business Use-Case Model, Business Modeling

1 Introduction

The practice of modeling the processes of an organization at the inception of – or continuously during – a software project has been adopted in many methods. Indeed, within a new information system development, being aware of the situation *as-is* is a fundamental prerequisite to define/align the system *to-be*. For such purpose, i* (i-star, [7]) has proven well; that is notably why it was adopted in Tropos [1] and I-Tropos [6] and why we suggest to include it into the RUP.

2 Objective of the Research

To address iterative development with Tropos, I-Tropos adapts the spiral life-cycle of the RUP in an i*-driven way. The approach followed by I-Tropos is nevertheless rather a revolution than an evolution for RUP practitionners since it is not UML-driven but based on a completely different set of artifacts. That is why, in [5], we have started to focus on mapping the i* semantics with the ones defined by the RUP business use-case model with the objective to fully capture the benefits of i* in the inherently iterative RUP. The gain for business analysts

would be to integrate i* benefits relying on RUP/UML business use-case syntax and semantics. This could ease the integration of i* in RUP even if the exact form it would take remains an open issue (see Section 5).

3 Scientific Contributions

3.1 UML Profile for i* Modeling

The research method applied to achieve our objectives firstly consisted in distinguishing groups of elements both within the ones defined by i* and the RUP/UML business use-case model. Elements considered here are the ones defined in the business modeling discipline of the *RUP knowledge base* [4] and provided into CASE tools like *Rational Rose* [2] or *Rational Software Architect* [3].

As presented in Table 1, three groups of elements have been distinguished within the i* ones: *Dependum Elements* (*DE*), *Actor Elements* (*AE*) and *Links* (*iStarLink*). Similarly, as presented in Table 2, three groups of elements have been distinguished within the RUP/UML business use-case model: *Inheriting from Use Case* (*IUC*), *Inheriting from the Actor* (*IA*) and *Links* (*UMLLink*). The groups have been made on the basis of the elements nature to ease the semantical mapping process.

Dependum Elements (*DE*)	*Actor Elements* (*AE*)	*Links* (*iStarLink*)
(Hard)goal	Actor	(Strategic) Dependency
Task	Position	Means-end
Resource	Agent	Decomposition
Softgoal	Role	Contribution
	Actor boundary	Actor association

Table 1. i* Elements to be Mapped

In order to compare i* elements and find best matches with UML ones, we have firstly compared the *DE* set with the *IUC* one, such as $DE \times IUC$. We indeed proceed through a carthesian product in order to compare the semantics of each pairs of elements issued of groups from the two modeling languages. However, no satisfying match was found for the *Softgoal* element so that we have further compared this element with the set *IA*. After having found the best possible matches for each element of the *DE* set, we have proceeded to a comparison of *AE* with *IA*, such as $AE \times IA$. However, no satisfying match was found for the *Actor Boundary* so that we further compared this element with the *IA* set. Finally, when this was achieved and the best possible match was found for each element in the set *IA*, we have compared the *iStarLink* set with *UMLLink* like $iStarLink \times UMLLink$.

(Hard)goal: *In a goal dependency, the depender depends on the dependee to bring about* ***a certain state of affairs in the world****.*

Inheriting from Use Case (IUC)	*Inheriting from the Actor (IA)*	*Links (UMLLink)*
Use Case	Actor	Unidirectional Association
Business Use Case (BUC)	Boundary	Dependency or Instanciates
BUC Realization	Business Actor	Generalization
Use Case Realization	Business Entity	Association
	Business Event	Aggregation
	Business Goal	Include
	Business Worker	Realize
	Control	Refine
	Domain	Extend
	Entity	Derive
	Interface	Package
	Table	
	View	

Table 2. Target UML Elements

Chosen Element: *Business Use Case.*
Rationale: Following the RUP knowledge base, *a Business Use Case (class) defines a set of business use-case instances in which each instance is a sequence of actions that a business performs that* ***yields an observable result of value*** *to a particular business actor.* The *Business Use Case* (*BUC*) element has been chosen because it is located at business (i.e., organizational) level such as the i* goal and yields an observable result of value.

Task: *In a task dependency, the depender depends on the dependee to carry out an activity. The dependum names a task which specifies how the task is to be performed, but not why. The depender has already made decisions about how the task is to be performed.*
Chosen Element: *Business Use Case Realization.*
Rationale: Following the RUP knowledge base, *a Business Use-Case Realization describes* ***how*** *business workers, business entities, and business events collaborate* ***to perform a particular business use case***. This corresponds to the purpose of the i* Task and is in line with the choice made for the (hard)goal element since we have selected the BUC at that stage.

Softgoal: *In a softgoal dependency, a depender depends on the dependee to perform some task that meets a softgoal. A softgoal is similar to a goal except that the criteria of success are not sharply defined a priori. The meaning of the softgoal is elaborated in terms of the methods that are chosen in the course of pursuing the goal.*
Chosen Element: *Business Goal.*
Rationale: Following the RUP knowledge base, *a Business Goal is a requirement that must be satisfied by the business. Business Goals describe the desired value of a particular measure at some future point in time and can therefore*

be used to plan and manage the activities of the business. This definition best corresponds to the purpose of the Softgoal.

Actor Boundary: Actor boundaries indicate intentional boundaries of a particular actor.
Chosen Element: *Package.*
Rationale: Following the RUP knowledge base, *a general purpose mechanism for organizing elements into groups. Packages may be nested within other packages.* Organizing elements into groups is precisely what we intend to do so we have selected this element for this purpose.

The UML Profile for i* Modeling. The result of our study is summarized in Table 3. The graphical notation is documented in [5].

i* element	Selected UML "rich" Use-Case Model Element
Goal	Business Use Case
Task	Business Use Case Realization
Resource	Business Entity
Softgoal	Business Goal
Actor	Business Actor
Position	Control
Agent	Actor
Role	Business Worker
Actor boundary	Package
(Strategic) Dependency	Dependency or Instanciates
Actor association	Generalization
Means-end	Include
Decomposition	Agregation
Contribution	Unidirectional Association

Table 3. i* Model Mapping

3.2 Discussion

This section highlights a number of open issues about the proposed mappings and justifies some choices made/compromises taken in a more global manner.

A functional goal is mapped to a business use-case; the rationale is based only on part of the definition of the latter element. One could argue that the definition also emphasizes that each instance of it is a "sequence of actions that a business performs", which, based on the corresponding i* definition, matches better with the notion of task. Nevertheless, a use-case realization is something even more concrete, so it makes sense to map tasks into something more detailed and tangible than in what goals are mapped. Still, a goal by itself does not have

any notion of a sequence of actions in it. Therefore, while the mapping is aligned to part of the definition, the compromise partly induces a semantic mismatch.

While the definition of a business goal as a desired value of a particular measure points to being a non-functional objective, the existence of such a desired value makes the goal objective and binary. Softgoals are subjective and can be achieved to some acceptable degree; a semantic distance is thus present.

When mapping an actor boundary to a package, an important semantic aspect of the model is potentially lost. An actor boundary is indeed not only a grouping of model elements. Goal refinement and analysis within that boundary is done from the point of view of the respective actor. Packages can only capture these semantics if additional constraints are included.

The two notations are consequently rather different and preserving semantics within such a mapping is a challenge. The form of integration is consequently of primary importance to higher the chances of adoption. If the purpose was to translate a particular model from i* to the RUP/UML business use-case model (or vice-versa), then some knowledge would typically be lost from the original model and other, new, knowledge would be required to be defined (manually) in the target model. This way both models could benefit since traceability between both analysis models is maintained. Also, additional advantage could come from the representation of the same problem using different modeling perspective. In [5], we point to the adoption of i* into the business modeling discipline of the RUP as only model relayed by a traditional use case model in the requirements discipline. This way, the (system) use case model would be built on the basis based on the lower-level (most operational) i* elements through a defined procedure. Other integration scenarios could nevertheless be envisaged and need to be studied to select the best possible option.

4 Conclusion

The first step in the research aiming to integrate i* within the RUP has been to study whether the RUP business use-case model provides elements that can be used as syntax with a semantic understanding that is compliant with the one associated to the i* ones. We have been able to find answers for each of them even if most often it was a matter of best possible compromise. Modeling in an i* fashion with the RUP/UML business use-case model syntax and semantics is thus possible; the graphical notation and an illustrative example are provided in [5]. The format of integration as well as reception of the new practice by the RUP community remain nevertheless open issues.

5 Ongoing and Future Work

In addition to the results presented so far, we highlight the fact that, over the years, i* modeling has been applied in collaboration with our research team in the context of multiple real-life industrial case studies to describe the situation "as-is". We notably refer to the development of a production management system

for a coking plant (2002-2007) and the development of a collaborative platform for outbound logistics actors (2008-2010). Some of these projects followed the RUP but i* modeling activities were applied "in parellel" rather than integrated into the unified development methodology. Also, i* was then applied with its traditional notation using custom CASE-tools. These cases can be used as basic material to study the alignment of i* models with business use-case ones.

Since our purpose is to integrate the i* approach (and thus its benefits) within a RUP/UML context we have to formally study the complementarity/overlap between the models to evaluate the best integration option. *Should we leave the business use case model into the RUP as a complementary/alternative view to i* models or should we use the i* model with the business use case model syntax only?* This can thus be the subject of an empirical evaluation through an ex-post analysis of the cases at disposal.

Next to this, if we want to favor industry adoption, we need to study sets of questions related to the practical adoption of i* by RUP practitioners. More precisely, we distinguish the following research questions:

- *To what extend are industry practitioners **able to** use the RUP syntax and associated semantic in an i* context?*
- *To what extend do industry practitioners **perceive** the benefits of i* modeling?*
- *To what extend are industry practitioners **willing to** change their habits to integrate i* modeling?*

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper.

References

1. J. Castro, M. Kolp, and J. Mylopoulos. Towards requirements-driven information systems engineering: the tropos project. *Inf. Syst.*, 27(6):365–389, 2002.
2. Terry Quatrani. *Visual Modeling with Rational Rose 2002 and UML*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 3rd edition, 2002.
3. Terry Quatrani and Jim Palistrant. *Visual Modeling with IBM Rational Software Architect and UML (The developerWorks Series)*. IBM Press, 2006.
4. Ahmad Shuja and Jochen Krebs. *Ibm®; rational unified process®; reference and certification guide*. IBM Press, first edition, 2007.
5. Yves Wautelet and Manuel Kolp. Mapping i* within uml for business modeling. In Joerg Doerr and Andreas L. Opdahl, editors, *REFSQ*, volume 7830 of *Lecture Notes in Computer Science*, pages 237–252. Springer, 2013.
6. Yves Wautelet, Manuel Kolp, and Stephan Poelmans. Requirements-driven iterative project planning. In María José Escalona Cuaresma, José Cordeiro, and Boris Shishkov, editors, *Software and Data Technologies*, volume 303 of *Communications in Computer and Information Science*, pages 121–135. Springer, 2013.
7. Eric Yu, Paolo Giorgini, Neil Maiden, and John Mylopoulos. *Social Modeling for Requirements Engineering*. MIT Press, 2011.

Aligning Data Warehouse Requirements with Business Goals

Alejandro Maté[1], Juan Trujillo[1], Eric Yu[2]

[1] Lucentia Research Group
Department of Software and Computing Systems
University of Alicante
{amate,jtrujillo}@dlsi.ua.es
[2] Department of Computer Science
University of Toronto
{eric}@cs.toronto.edu

Abstract. According to the Gartner Group, over 70% of Business Intelligence (BI) projects fail. Among the reasons are the different languages employed by IT and business people and the necessity of a long-term BI plan describing the goals of the organization. In current practice, when building the data warehouse (DW), strategic plans are rarely considered. In this paper, we propose a method to align the business plan with DW requirements analysis. By aligning the DW, we (i) validate the correctness of each decision makers' goals, (ii) ensure that their decisions and the DW contribute towards organization goals, and (iii) provide a long-term unified BI strategy. We instantiate this alignment by combining i* for DWs with strategic business models.

Keywords: Business Intelligence, business plan, data warehouses, alignment, BIM

1 Introduction

Data Warehouses (DW) integrate numerous heterogeneous data sources in multidimensional structures in support of the decision-making process. In order to be successful, recent DW development approaches, such as [5], include a requirement analysis step based on goal models. During this step, the requirements of individual decision makers, who are the users of the Business Intelligence (BI) system, are gathered by means of questionnaires and interviews.

The main drawback of the techniques used during the requirements step is that they obtain only a partial view of the problem from each decision maker. Moreover, these partial views may not always be aligned with the business plan, which, together with the gap between IT and business people, makes it difficult to validate the goal models. In turn, the lack of alignment between individual decision makers and the business plan is translated into a lack of long term enterprise BI strategy, which is one of the key factors to successfully apply BI. This problem arises whenever business goals are not considered during the requirements analysis step. For example, a fictitious car dealer company EURent

pursues the goal of "Be positioned within the top 10 car dealers". EURent Sales Manager is considering how to improve the "Car Sales" process by "Increasing the margin of benefit per sale". While this particular goal improves the results of the sales process, it does not contribute to the increasing number of cars sold.

In this paper, we propose to tackle this problem by aligning the business strategy with current BI and DW goal models. Therefore, we ensure that our BI-enabled decision making is consistent with the business strategy. First, we elaborate a strategic goal model from the business plan to formalize business goals and trace business indicators. Then, we align and relate decision makers' goals with business goals. Therefore, we ensure that all the decision makers are contributing towards the overall goals of the enterprise, and identify business goals that are being overlooked by decision makers.

The remainder of this paper is structured as follows: Section 2 presents the related work in DW requirements. Section 3 describes the alignment process and the main contributions of our work. Section 4 presents the conclusions. Section 5 describes the ongoing and future work in this area.

2 Related Work

Until now, most of the attention in DW requirements analysis has been focused on requirements elicitation. Current techniques [2, 5–7] are focused on gathering requirements by means of interviews and questionnaires. However, although it has been stated that DW requirements can rarely be gathered correctly and comprehensively from individual decision makers [7], only in [2] do the authors consider organizational modeling in the requirements step. Unfortunately, the focus of organizational modeling in [2] is to identify the information stored in business processes, rather than obtaining business goals. Therefore, these approaches do not ensure that decisions being taken are aiming to provide a benefit for the organization according to the business plan, nor they do ensure that all the organization's goals have been considered in the decision making process.

3 Scientific Contributions & Tool

In this section we present the main contributions of our work, as well as the current tool support for our approach. Our contributions are described in the following order: (i) a method to perform the alignment between DW requirements and business goals, (ii) a set of mappings to instantiate the alignment method, by using i* for DWs [5] for modeling decision makers' goals, and the Business Intelligence Model (BIM) [1] for modeling the business strategy, and (iii) show how these mappings and models can be implemented in a tool.

First, the alignment method is shown in Figure 1. The alignment process starts by modeling decision makers' goals by means of techniques used in current DW approaches. As a result, an i* for DWs model is created for each decision maker. In parallel, or after DW requirements have been obtained, the business

strategy model is created by using the information from the business plan. Business plans include information regarding business goals, the objectives associated with each goal, and may also include information regarding business processes. Once both models have been obtained, we proceed to the alignment step. First, we align the concepts used to model decision makers' goals with those used to model business goals, in order to ensure that we correctly relate individual goals with business goals. Then, in collaboration with a domain expert, we align each concrete decision maker goal to a specific business goal. Finally, we analyze the alignment and perform changes as necessary.

Since currently there is no standard for modeling decision makers' goals nor business strategies, they can be instantiated according to different frameworks, thus leading to different results. Therefore, in order to instantiate the method, it is necessary to perform an ontological mapping between the concepts of the specific frameworks being used. In our case, we make use of i* for DWs in order to model decision makers' goals, and BIM for modeling the business strategy. Thus, we analyze the definitions and characteristics of each concept included in these metamodels and align them to ensure their correct use. Due to space constraints, we provide only a brief overview of the metamodels, and focus specifically on aligned elements.

First, i* for DWs [5] is an extension of i* that includes several types of actor goals to describe the decision making process rationale: Strategic Goals, Decision Goals, and Information Goals. Strategic Goals are goals with the highest level of abstraction, such as "Attract new customers", while Decision and Information Goals are included to describe the steps required in the decision process to achieve the Strategic Goals. We formally describe a Strategic Goal as follows: A strategic goal S_i describes a change from a current situation into a better one, as seen by *one* decision maker DM_j. The change described by S_i must be related to an objective O_k of *one* Business Process BP_m, e.g. "Sales". Therefore, it is said that S_i *improves* BP_m. If S_i is achieved, it causes an immediate benefit B_p for the organization, that may or may not be measurable. Finally, i* for DWs also includes lower level abstraction elements that realize Information Goals and define the structure of the DW: Contexts and Measures. Contexts describe

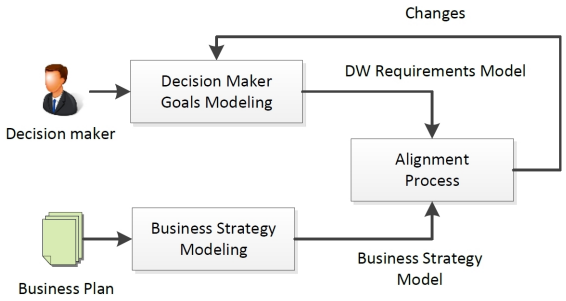


Fig. 1. Steps involved in the alignment method

concepts involved in BP_m, such as "Customers", while Measures are quantitative values v that evaluate the performance of BP_m, such as "Quantity Sold".

Second, in order to create the business strategy model, we use BIM [1, 4]. BIM includes four core concepts that allow us to formalize the elements within the business plan: Intentions (business goals), Situations, Indicators, and Processes (see [4]). An Intention defines a desired situation for the enterprise, as defined by one or more persons of authority $A = \{A_i, ..., A_n\}$ in the business plan, where A usually is a subset of the decision makers. Intentions can be classified as Strategic (long term), Management (medium term), and Operational (short term), and are realized by one or more Processes, $BP_i, ..., BPn$. An example of a Strategic Intention of the organization EURent is "To be a Premium Brand car rental company". Next, Situations describe internal or external factors, such as "Economic Crisis", that may affect Intentions or Processes, while Indicators measure the performance of a Process or Intention. Indicators include a target value t to be achieved, a threshold th that separates average from bad performance, and a worst value w that describes the worst performance. Finally, Processes represent Business Processes, such as "Sales" or "Rentals".

According to the descriptions provided in the i* for DWs and BIM proposals, Decision and Information Goals, and Contexts from i* for DWs cannot be aligned with any element in BIM. The main reason is that BIM focuses on representing the business strategy and does not consider elements specific to the individual decision making process, such information requirements or actors.

Next, Business Processes and Measures from i* for DWs can be easily aligned with BIM elements. First, Business Processes refer to the same concept as Processes. Second, Measures provide a value v_i that measures the performance of a business process. These values are one of the basic elements used to calculate Indicators that monitor elements in the business strategy. A Measure can be related or transformed into an Indicator I_j if (i) v_i is used in the calculus of I_j and (ii) the rest of the values t_j, th_j, and w_j are defined.

Finally, Strategic Goals can be aligned with Intentions as follows: according to our formalization, achieving a particular decision maker strategic goal

Table 1. Elements aligned between i* for DWs and BIM and their characteristics

i* for DWs	BIM	Details i* for DWs	Details BIM
Business Process	Business Process	Not detailed, Strategic Goals improve its results	May be detailed, Realizes goals
Strategic Goal	Strategic Intention	Future, Focused, Long-term, Qualitative or Quantitative, Improves business process	Future, Focused, Long-term Qualitative, Measured by KPI
Measure	Indicator	Measures the performance of a Business Process without target values	Future, Time-Targeted, Long or Short-term, Measures Goal and Process performance

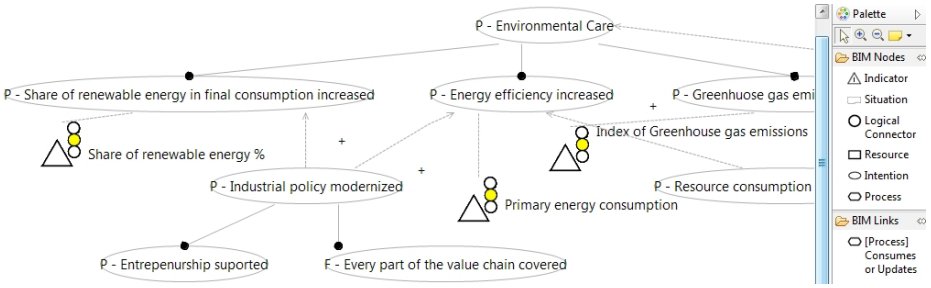


Fig. 2. Business strategy editing using the Lucentia BI Tool

S_i *improves* a process BP_m. Thus, this improvement helps the organization to achieve a business intention I_j, realized by BP_m. There are, however, two limitations to automate this alignment. First, S_i may improve a specific objective O_k that does not have an impact on I_j. For example, improving the "Car Sales" process by "Increasing the margin of benefit" does not have an impact on "Be positioned within the top 10 Car Dealers". Second, in practice, not every business plan or strategy provides references to the business processes that realize each intention. Thus, it is necessary to collaborate with a domain expert in order to identify if the particular goals are aligned with the business intentions or not.

The alignment is summarized in in Table 1, along with the informal descriptions provided by the frameworks for each element.

After aligning the concepts, the last step is aligning the elements within the goal models obtained from decision makers with the business strategy, thus obtaining a combined model that covers both individual as well as organizational views. We have successfully tested our approach in an experimental case study, creating a business strategy from a business plan and interviewing experts in information analysis to obtain decision makers' goals. However, due to space constraints, the case study will be described in a future extended version.

Our approach is supported by our tool, the Lucentia BI suite, which allows us to model (i) user requirements by means of i* for DWs, (ii) business strategies by means of a particular implementation of BIM, and (iii) a trace metamodel, described in [3], that allows us to materialize the alignment between user requirements and the business strategy. The tool is based on Eclipse and includes a set of modules, each designed to support an specific task. An screenshot of the tool can be seen in Figure 2.

4 Conclusions

In this paper we have presented an alignment between the business strategy, modeled using BIM, and DW requirements, represented by i* for DWs. Our process results in an alignment that allows us to (i) validate DW requirements according to the business strategy and identify non-aligned goals, (ii) provide a

long-term BI strategy to be pursued, including what information is being used by the organization to support each goal, (iii) identify the different decision makers participating in a business goal, thus providing awareness, and (iv) evaluate if decision makers are being successful by analyzing the values of indicators related to business goals.

While extending i* to consider all the elements within the business strategy would overcomplicate the model, our initial applications have shown that it can be combined with other models in order to capture both the particular viewpoint of each decision maker as well as the overall strategy of the organization.

5 Ongoing and Future work

The current ongoing work is focused on making it easier for businesses to apply our proposal. In order to achieve this, we plan to semi-automate the process of obtaining an strategic model directly from the business plan. Therefore, we are analyzing the viability of defining a series of pattern-based transformations in order to save time and costs.

In the medium-term we plan to apply our approach to a real case study and evaluate the results. Since then structure of business plans may vary from one organization to another, we plan to minimize the impact of this variability by using the standard Business Motivation Model, proposed by the Object Management Group group.

Acknowledgments This work has been supported by the MESOLAP (TIN2010-14860) and GEODAS-BI (TIN2012-37493-C03-03) projects from the Spanish Ministry of Education and the Spanish Ministry of Economy and Competitivity. Alejandro Maté is funded by the Generalitat Valenciana (ACIF/2010/298).

References

1. D. Barone, E. Yu, J. Won, L. Jiang, and J. Mylopoulos. Enterprise modeling for business intelligence. *The Practice of Enterprise Modeling*, pages 31–45, 2010.
2. P. Giorgini, S. Rizzi, and M. Garzetti. GRAnD: A goal-oriented approach to requirement analysis in data warehouses. *Decision Support Systems*, 45(1):4–21, 2008.
3. A. Maté and J. Trujillo. A trace metamodel proposal based on the model driven architecture framework for the traceability of user requirements in data warehouses. *Information Systems*, 37(8):753 – 766, 2012.
4. A. Maté, J. Trujillo, and J. Mylopoulos. Conceptualizing and specifying key performance indicators in business strategy models. *ER'12*, pages 282–291, 2012.
5. J.N. Mazón, J. Pardillo, and J. Trujillo. A model-driven goal-oriented requirement engineering approach for data warehouses. *Advances in Conceptual Modeling–Foundations and Applications*, pages 255–264, 2007.
6. N. Prakash and A. Gosain. Requirements driven data warehouse development. In *CAiSE Short Paper Proceedings*, pages 13–17, 2003.
7. R. Winter and B. Strauch. A method for demand-driven information requirements analysis in data warehousing projects. In *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*, pages 9–pp. IEEE, 2003.

Using *i** to Represent OSS Ecosystems for Risk Assessment

Claudia Ayala[1], Xavier Franch[1], Lidia López[1], Mirko Morandini[2], Angelo Susi[2]

[1]Software Engineering for Information Systems Research Group (GESSI)
Universitat Politècnica de Catalunya (UPC)
Barcelona, Spain
{franch|cayala|llopez}@essi.upc.edu
[2]Fondazione Bruno Kessler, Center for Information Technology,
via Sommarive 18, 38123 Povo, Trento, Italy
{morandini|susi}@fbk.eu

Abstract. Open Source Software (OSS) is a strategic asset for organisations thanks to its short time-to-market, the opportunity for a reduced development effort and total cost of ownership, and its customization capabilities. OSS-based solutions include projects that are developed and co-evolve within the same organisation, OSS communities, companies, and regulatory bodies, forming an articulated strategic business ecosystem. The adoption of OSS in commercial projects leads to numerous challenges in the wide spectrum of available OSS solutions and risks emerging from the intrinsic structure of an OSS project. In this position paper we devise the use of *i** models for understanding the strategic perspective of OSS ecosystems, representing actors, intentional dependencies and responsibilities. We argue that these models can play a crucial role in the analysis of organisational risks inherent to OSS component adoption and in the definition of risk mitigation activities.

Keywords. OSS, iStar, Software adoption, risk, goal-oriented requirements engineering.

1 Introduction

The strategic importance of Open Source Software (OSS) technologies in the development of commercial software is growing steadily. Gartner recently highlighted[1]: by 2016 OSS will be included in 95% of all commercial software packages. In spite of this trend, IT companies and organizations are still facing difficulties and challenges when they decide to adopt OSS solutions. It stands that OSS is about freedom and choice, but freedom and choice introduce risks[2].

OSS components integration in the development of complex software systems has became a popular way of OSS adoption. However, despite the potential benefits of

[1] Understand the Challenge of Open-Source Software. Gartner Reports, September 2012.
[2] Critical Strategies to Manage Risk and Maximize Business Value of Open Source in the Enterprise. Gartner Reports, June 2011.

reusing OSS components, such integration involves several risks and challenges. As a result, traditional project and risk management strategies should be put forward; specially for assessing the wide, often complex, interactions of the diverse actors related with the internal and external development, maintenance and evolution of the OSS component and the context where the OSS component will be integrated. Thus, two important viewpoints should be considered and reconciled: the OSS project ecosystem and the adopting organization ecosystem. On the one hand, the OSS project ecosystem. comprises not only the developers, but the whole community, including users, regulatory bodies, and companies involved with the project (if any). It also covers technical support, marketing and possible financial aspects (including the business model(s) behind the project). On the other hand, the adopting organization ecosystem comprises the technical and business issues of the project where the OSS component will be integrated. Both ecosystems together could be considered as the OSS-based ecosystem. In this complex setting several questions emerge, e.g.:

- Which viewpoints should be considered for assessing the OSS-based ecosystem in order to ensure a smooth integration and evolution of the OSS component?
- How to secure that specific properties of an OSS do not harm business models and their underlying business strategies?
- How to implement a systematic approach towards understanding and representing dependencies that involve OSS components, for assessing possible risks?

We believe that the answer to these questions requires a clear understanding of the ***OSS-based ecosystems from a strategic perspective***, with the identification of strategic dependencies (not just related to software component dependencies) as a means to identify risks coming from the adoption of OSS components and to design risk mitigation strategies along the lifetime of a software product.

Along this line the paper introduces an approach, currently explored in the context of the European project RISCOSS (RISks and Costs in Open Source Software adoption), that promotes the use of *i** to understand the strategic perspective of OSS-based ecosystems.

2 Objectives of the research

In the RISCOSS project we envision methods and tools for supporting the analysis of risks and costs that organizations need to evaluate for deciding the integration of OSS components in the development of software products.

The processes for the adoption of commercial off-the-shelf (COTS) components are well established in many companies. Many medium and large companies follow strict guidelines for risk analysis, cost estimation, and contract agreement. Such agreements are typically based on the principles of liability and confidence, in exchange of a single or recurrent fee. However, when it comes to OSS components, the situation is different. On the one side, there is access to the source code, making possible (depending on license implications) to use and customize the component without any contract agreements and payment authorization [1]. This opens the possibility for a less bureaucratic but uncontrolled use of OSS components without a thorough anal-

ysis of risks and undesirable implications, as evidenced in industrial surveys [2]. On the other side, the own characteristics of the OSS project ecosystem are highly different from that of traditional software. The community behind the OSS component is not generally driven by a single business goal, the motivations and objectives of the contributors are manifold, the software is provided without quality of service agreements and the community cannot formally commit on the future roadmap.

Consequently, to evaluate risks and opportunities of OSS components adoption, it would be crucial to further understand the strategic perspective of the OSS project ecosystem in order to evaluate its adequacy to the strategic perspective of the adopting organization (i.e., the adopting organization ecosystem).

Important risks related to the OSS project ecosystem [1] [2], could be for instance: the lack of a roadmap and/or ownership of the project, unclear liability/responsibility, and bug fixing time. From the point of view of the adopting organization, liability and support can sometimes be commissioned, to a certain degree, to specialised companies, thus de facto forcing the component to go through the COTS adoption process. However, this is not always possible and often not a favourable solution, both from the point of view of costs and additional risks. Moreover, the uncoupling of companies, developers and software gives rise to issues with which traditional risk analysis processes are not able to cope, as evidenced in various empirical studies [4].

In this context, our first objective is to support the risk assessment processes of OSS adoption by using *i** models as a basis for the analysis of the strategic perspective of the OSS-based ecosystem that involves the assessment of the OSS project ecosystem and the adopting organization ecosystem.

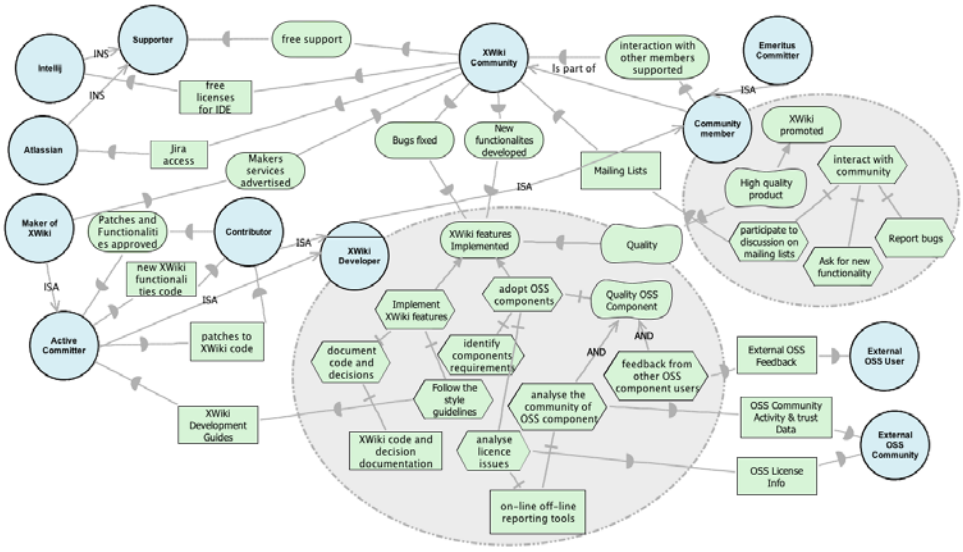


Figure 1: An excerpt of the strategic diagram representing the XWiki project.

Figure 1 displays an excerpt of the strategic diagram for an OSS project called XWiki (www.xwiki.org), which is one of the case studies of the RISCOSS project. It shows the structure and responsibilities related with the XWiki community, composed by developers (contributors and active committers) and other community members. Note that some of the roles are not "assigned", i.e. there is no formal commitment to these roles. E.g., users of XWiki become community members when they send bug reports and feature requests or discuss on forums and mailing lists. The XWiki project development is mainly driven by the company XWiki SAS (www.xwiki.com), which is also part of the model in order to explicitly show the influence of the company on the project and to understand the influence of its business strategy.

These models could be a valuable tool for the adopting organizations, for having an overview on the OSS project ecosystem and evaluating the risks implied in a potential adoption. For example, lack of ownership could be evaluated by analysing the companies and individuals that have prominent (business) dependencies and responsibilities in a project, while the lack of a roadmap could be investigated by analysing the structure of the community, identifying the amount of distributiveness, the presence of heroes, and the roles and business objectives of the companies involved.

We plan to provide specific guidelines and measures for a qualitative assessment of an OSS project ecosystem and its related risks, to support the decision process and to provide an entry point for enacting proper mitigation activities.

3 Scientific contributions

In this work we envisage several scientific contributions that are mainly related to the use of *i** for supporting the assessment of risks in OSS-based ecosystems.

Patterns of ecosystems and organisations. A first research challenge is the possibility to extract and represent OSS-based ecosystem patterns (at different levels of abstraction). These patterns would allow abstracting the roles, goals and dependencies in the OSS project ecosystem and in the adopting organization ecosystem; and identifying schemas that have the property to be more prone to risks or to be particularly useful to implement organisational risk mitigation strategies. They should identify and highlight properties of an ecosystem, such as ownership and leadership in the structure of the community, its stability (distributiveness, centralism, presence of heroes), the role and involvement of companies in the project (e.g., technical support, financial promotion, developer contribution, community support and influence), as well as their business model.

To do so, we have identified several dimensions to classify the entities involved in such ecosystems:

- Role: producer, consumer, community.
- Setting: industrial (large, medium, small), academia, public administration.
- Business strategy: from full OSS collaboration to OSS exploitation.
- Business process: adoption, migration, consolidation, and improvement.

Each data point determined by these dimensions provides a scenario that may be analyzed and characterised by different patterns. In RISCOSS we are currently ap-

proaching several of these scenarios by analysing 5 use cases that cover situations from very limited OSS implication in the business strategy to a full collaborative approach. Such real cases will help to establish a solid evidence for the design of our patterns. For instance, we have a large industrial company that produces highly reliable software products and aims to integrate an OSS component in its software product line. Its business strategy regarding OSS is to just exploit the component functionality (without involvement in the OSS project) and does not have interest to change processes. On the other hand, we have also the case of a medium-sized company, whose business strategy relies entirely on OSS adoption and development and therefore adapted its processes and the whole organization to such an approach.

Level of abstraction in the models. Another important issue is related to the need of representing the ecosystems at several levels of detail, at both class and instance level, to facilitate the reasoning about the model, focusing on the high level structure of the ecosystem or going into further details, with a particular focus on OSS-specific actors such as individuals, community groups, etc. In line with previous experiences in other settings [3], in order to use *i** models as a communication means, we need to keep the models as simple as possible, so they could be understood by all the industrial partners of the RISCOSS project. So far, we are using general actors without classifying them except for those that clearly are agents. Also we limit the use of soft goals to the most fundamental ones. The number and relationships of actors has driven us to adopt a third kind of model, the Strategic Actor Diagram as proposed by Leite et al. [4], which gives a useful perspective on the system.

Guidelines for the specification of the models and repositories. To support the specification of the OSS-based ecosystems, a clear process is needed to specify a set of *i** diagrams. We have taken as a starting point the R*i*SD methodology for SD diagrams [5], and we plan to refine it and to create similar versions for the other types of *i** diagrams. A shared or company-internal repository of such models could be implemented to allow the evaluation of the structure of the OSS-based ecosystem, based on organisational patterns. Thanks to this repository, analysts could get an overview of the OSS project ecosystem and adopt analysis guidelines to identify risks and to manage them for getting pursuable criteria for OSS adoption decisions.

New modelling concepts. We are also considering the need of enriching the set of *i** modelling concepts on the basis of specific characteristics of the ecosystems and of domain risks [6]. Concerning the ecosystems, there could be a need for concepts able to express aggregation between actors for a direct representation of teams or communities. Also, some specification relationships between actors may be expected to emerge. Moreover, in a community-driven environment, central *i** concepts such as *delegation* and *responsibility* need to be re-discussed. In OSS, the strict concept of *delegation* changes to a more relaxed concept of *expectation* and *observance* of *norms*. Responsibility is scattered and can often not be clearly identified, and must thus often be taken over by the companies adopting the software for their products. Moreover, in this context many roles are defined by access rights and by own interest and can dynamically change. Concerning risks, we plan to follow a strategy similar to the one used in Nomos [7], namely incorporating new concepts for the representation of risk, of the impact that the risk has on the structure of the organisation and of pos-

sible mitigation activities. Available *i** risk modelling approaches such as [8], defining risks, events, affected assets and treatments, will be taken into account for this as well.

4 Conclusions, ongoing and future work

In this paper we described one of the main objectives of the European project RISCOSS that is to support decision making related to the assessment of risks in the integration of OSS components. We envision the use of *i** models to capture the intentional aspects that drive the OSS project ecosystem as well as the adopting organization ecosystem, in order to analyse both points of view and their potential risks.

Based on the analysis of five adopting organizations represented by the industrial partners of the RISCOSS project, we are currently: applying existing guidelines for the specification of *i** models, identifying elements and relationships that may represent potential patterns or new modelling concepts. Furthermore, we are performing systematic literature reviews on OSS ontologies, OSS ecosystems and OSS risk management with the purpose of developing an ontology to be linked to the *i** core. We plan to use some foundational ontology (e.g., UFO, DOLCE) to connect these two worlds, aligning with ongoing research on the semantic meaning of *i** constructs [9].

Acknowledgements

This work is a result of the RISCOSS project, funded by the EC 7th Framework Programme FP7/2007-2013 under the agreement number 318249.

References

[1] Morgan, L. & Finnegan, P. Open innovation in secondary software firms: an exploration of managers' perceptions of open source software SIGMIS Database, ACM, 2010, 76-95.
[2] Hauge, Ø.; Cruzes, D.; Conradi, R.; Velle, K. S. & Skarpenes, T. A. Risks and Risk Mitigation in Open Source Software Adoption: Bridging the Gap between Literature and Practice OSS, 2010, 105-118.
[3] Carvallo J.P., Franch X.: On the Use of *i** for Architecting Hybrid Systems: A Method and an Evaluation Report. PoEM 2009.
[4] Leite J. et al.: Understanding the Strategic Actor Diagram: an Exercise of Meta Modelling. WER 2007.
[5] Franch X. et al.: Systematic Construction of *i** Strategic Dependency Models for Socio-Technical Systems. IJSEKE 17(1), 2007.
[6] Franch X., Maté, A., Trujillo J.C., Cares C.: On the joint use of i* with other Modelling Frameworks: a Vision Paper. RE, 2011.
[7] Alberto Siena, Ivan Jureta, Silvia Ingolfo, Angelo Susi, Anna Perini, John Mylopoulos: Capturing Variability of Law with Nómos 2. ER 2012: 383-396
[8] Asnar, Y; Giorgini, P., Mylopoulos, J. : "Goal-driven risk assessment in requirements engineering" Requirements Engineering. 16(2), 101-116. 2011.
[9] Guizzardi R., Franch X., Guizzardi G.: Applying a Foundational Ontology to Analyze Means-End Links in the *i** Framework. RCIS 2012.

Designing Secure Socio-Technical Systems with STS-ml

Elda Paja[1], Fabiano Dalpiaz[2], and Paolo Giorgini[1]

[1] University of Trento, Italy – {elda.paja, paolo.giorgini}@unitn.it
[2] University of Toronto, Canada – dalpiaz@cs.toronto.edu

Abstract. A Socio-Technical System (STS) is an interplay of humans, organizations and technical systems. STSs consist of interacting actors, which depend on one another to achieve their objectives. In previous work, we have proposed STS-ml, a security requirements modelling language (using *i**-like primitives such as actor, goal, delegation) for the design of secure STSs. STS-ml represents security requirements as constraints over the interactions (goal delegation and document exchange) among actors in the STS. In this work, we present the current version of STS-ml, which introduces further modelling primitives as well as sophisticated reasoning mechanisms to detect conflicts in security requirements.

1 Introduction

Socio-Technical Systems (STSs) are an interplay of social (human and organisations) and technical subsystems, which interact with one another to reach their objectives, making an STS a network of social relationships [1]. Each subsystem is a participant of the STS, acts according to its business policy (it is autonomous), and interacts with others. When relying upon others for carrying out tasks and manipulating information, a participant would like its own security requirements to be fulfilled. For example, if a buyer sends its personal data to a seller, the buyer may require the data to be used only for shipment purposes.

To deal with security in the early phases of STS design, we have previously proposed STS-ml [2] (Socio-Technical Security modelling language), an actor- and goal-oriented security requirements modelling language. STS-ml is based on the idea of relating security requirements to *interaction*. As such, the language allows stakeholders to express *security needs* over their interactions to constrain the way interaction is to take place, as in the buyer/seller example above.

STS-ml specifies security requirements as *social commitments*, promises with contractual validity made by an actor to another. One actor commits to another that, while delivering some service, it will comply with the required security needs. In the example above, a security requirement is that the seller commits not to disclose personal data to other parties.

These specifications guide the design of a STS that satisfies the security requirements. However, in certain cases, the specification may be *inconsistent*, i.e., one or more requirements might be conflicting. It is, thus, not possible to implement a STS that satisfies all requirements of the specification.

Coping with such conflicts at requirements-time avoids developing a non-compliant and hard-to-change system. We propose to rely on automated reasoning to identify and resolve these conflicts. This choice is justified by our gathered evidence [5] that requirements models are large and that even skilled analysts would be unable to identify all the conflicts in a model.

2 Objectives of the research

We aim at creating a framework that supports the identification of inconsistencies (conflicts) among security requirements in a STS-ml specification. We focus on two types of conflicts: (1) *among security requirements*: two or more security requirements cannot be implemented by the same system. For instance, access to some information may be granted from one stakeholder, and prohibited from another. The different authorisations are conflicting, and one of them needs to be relaxed and adapted, perhaps by negotiating the needs of the authorisee; and (2) *between actors' business policies and security requirements*: an actor's policy may specify that some information shall be accessed, while no authorisation is granted by the information owner. We provide examples in Sec. 3.1.

While analysts may detect some conflicts by looking at the graphical models, others are harder to spot—especially in large models—and require computer-aided support. To such extent, we have formalised the semantics of the STS-ml primitives and that of its *security requirements*. This effort enables us developing automated reasoning techniques for the identification of conflicts. Our longer-term objective is to support the resolution of conflicts too.

3 Scientific contributions

STS-ml is an actor- and goal-oriented security requirements engineering method. It is composed of the modelling language and its support tool STS-Tool [3]. The contributions of this research are as follows:

- a revised version of the STS-ml language, including a wider set of security requirements;
- a formal framework to identify conflicts among security requirements, as well as among actors' business policies and the security requirements they are required to comply with;
- an implementation of the framework in disjuctive Datalog. STS-Tool supports the graphical visualisation of conflicts in STS-ml diagrams.

3.1 The STS-ml framework for security requirements engineering

STS-ml has been first proposed in [2], here we present the current version of STS-ml. STS-ml includes high-level organisational concepts such as actor, goal,

[3] `http://www.sts-tool.eu`

delegation, etc. Security requirements in STS-ml models are mapped to *social commitments* [4]—contracts among actors—that actors in the STS shall comply with at runtime. STS-ml modelling consists of three complementary views of the same model, namely *social*, *information*, and *authorisation view* (see Fig. 1), so that different interactions among actors can be analysed by concentrating on a specific view at a time. Interview consistency is ensured by STS-Tool [4].

Example 1 (Travel Planning). A tourist wants to organise a trip using a Travel Agency Service (*TAS*). *TAS* allows end users to read about various destinations, book flights and hotels, hire a car, etc., and uses the *Amadeus flight service* to book flight tickets. To book hotels, the *Tourist* has chosen to directly contact the *Hotel* himself, without interacting with *TAS*.

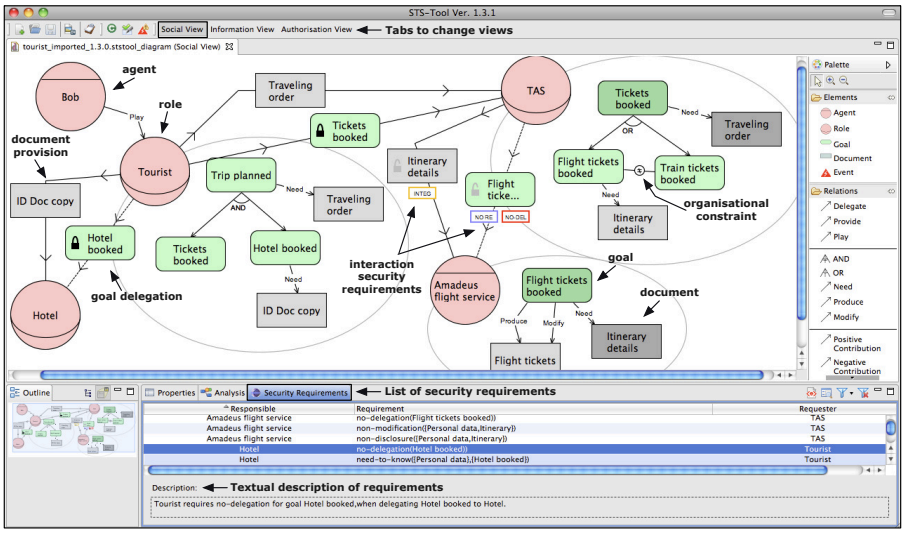
The *social view* (Fig. 1a) represents actors as intentional and social entities. Actors are intentional as they aim to attain their goals, and they are social, for they interact with others by delegating goals and exchanging documents. STS-ml supports two types of actors: agents—concrete participants, and roles—abstract actors, used when the actual participant is unknown. In our example, we represent the *TAS* as a role, and the *Amadeus flight service* as an agent, as it refers to a specific flight service. Actors may possess documents, they may *use*, *modify*, or *produce* documents while achieving their goals. For instance, *Tourist* wants to achieve the goal *trip planned*, for which it needs to both have *tickets booked* and *hotel booked*. To book the hotel it needs document *ID Doc copy*.

The *information view* (Fig. 1b) gives a structured representation of actors' information and documents, and the way they are interconnected. Information can be represented by one or more documents (through TangibleBy), and on the other hand one or more information entities can be part of some document. For instance, information *Personal data* is represented by both *ID Doc copy* and *Flight tickets* documents. We keep track of how information and documents are interconnected to identify which information actors manipulate, when they *use*, *modify*, *produce*, or *distribute* documents to achieve their goals. We take a pessimistic approach assuming that whenever a document is altered, the information it makes tangible is also altered. For instance, the *Amadeus flight service* modifies information *Personal data* when modifying document *Flight tickets*.

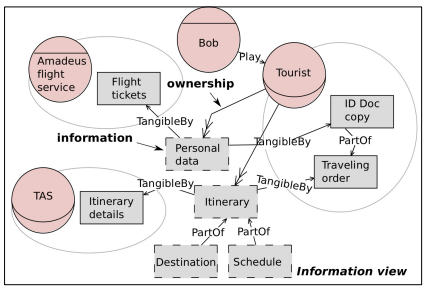
The *authorisation view* (Fig. 1c) shows the authorisations actors grant to others over information, specifying which operations they are allowed (prohibited) to do, for which goals (scope), and whether authorisation can be further transferred or not. For instance, *Tourist* authorises *TAS* to use (U selected) information *Personal data* and *Itinerary* in the scope of the goal *Tickets booked* granting a transferrable authorisation (authorisation's arrow line is continuous). Through its three views, STS-ml supports different requirements types:

– *Business policies* are expressed by specifying actors, their goals, delegations, document provisions, and how actors manipulate documents to fulfil goals. In a nutshell, they are represented by an actor's goal model.

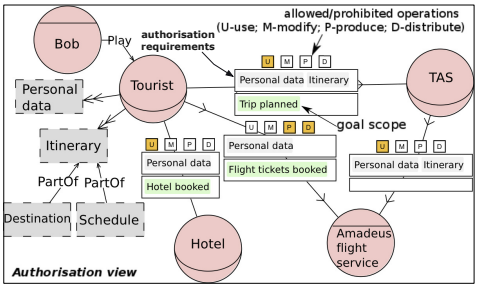
[4] `http://www.sts-tool.eu`



(a) Social view



(b) Information view



(c) Authorisation view

Fig. 1: Multi-view modelling for the travel planning scenario

– *Interaction (security) requirements* are security constraints on goal delegations and document provisions. STS-ml supports *non-repudiation*, four types of *redundancy*, *no-redelegation*, *availability* and *integrity-of-transmission* interaction security requirements. For instance, *TAS* requires the *Amadeus flight service* not to repudiate the delegation of goal *Flight tickets booked*, and also not to redelegate this goal. The *Amadeus flight service*, on the other hand, requires that the integrity of document *Itinerary details* is preserved.
– *Authorisation requirements* determine which information can be used, how, for which purpose, and by whom; at the same time they define also prohibitions over the same information. In this category, STS-ml supports *non-reauthorisation*, *need-to-know* of information, *non-usage*, *non-modification*,

non-production and *non-disclosure* of information. While authorising *TAS* to use (U selected) information *Personal data* and *Itinerary* in the scope of the goal *Tickets booked*, the tourist is prohibiting *TAS* to perform the rest of operations, requiring *non-modification*, *non-production* and *non-disclosure* of the given information. Moreover, since authorisation is limited to a goal scope, this expresses a *need-to-know* requirement over information, demanding it to be used only for the specified scope;

– *Organisational constraints* determine constraints on the adoption of roles and the uptake of responsibilities. STS-ml supports *separation* and *binding of duties* both over roles and over goals. For instance, a separation of duty is expressed between goals *Flight tickets booked* and *Train tickets booked* requiring *TAS* not to ever achieve both these goals.

Together, interaction requirements, authorisation requirements, and organisational constraints are the security requirements of STS-ml. A STS-ml model is consistent if the actors' business policies comply with the security requirements.

3.2 Detecting conflicts in security requirements

We are concerned with the verification of two types of inconsistencies that relate to security requirements: (i) identifying conflicts among security requirements, that is, verifying whether the simultaneous specification of different security requirements brings up conflicts; and (ii) identifying conflicts raised by the specification of the security requirements with respect to an actor's business policy. For each of these two categories, we identify conflicts in our running example, and provide insights on how they could be fixed. Here we provide just the intuition behind the identification of these conflicts. The formal framework, its implementation in disjunctive Datalog, and experimental results from an industrial case study on eGovernment are presented in a technical report (see [3]).

The first challenge is to obtain a set of security requirements that is consistent, which enables us to build a secure socio-technical system (that violates no security requirement). The focus of STS-ml is mainly on information security; thus, we have begun by tackling authorisation conflicts. Detecting conflicts among other types of security requirements is still work in progress.

For instance, the *Tourist* does not authorise the *Amadeus flight service* to use his/her *Personal Data*, while *TAS* authorises the *Amadeus flight service* to use *Tourist*'s *Personal Data*. This situation represents an authorisation conflict for the *Amadeus flight service*. Specifically, this conflict relates to the "use" operation. Our framework supports also conflicts about modification, production, distribution, and *authorisation transferibility* (i.e., an actor is granted authorisation to perform operations but not that of further distributing the authorisation).

The second challenge is to verify if one or more security requirements conflict with the private business policy of an actor (as dictated by its goal model). For instance, suppose the *Hotel*'s business policy requires handling all bookings by means of a reservation system service. When *Tourist* specifies a *no-redelegation* security requirement over the delegation of the goal *Hotel booked*, there is a

conflict between what the security requirement demands the Hotel to do, and what its internal requirement dictates. Let us show another example of this kind of conflict. The *Amadeus flight service* modifies *Flight tickets* when achieving goal *Flight tickets booked*. *Flight tickets* makes tangible tourist's *personal data*, for which no authorisation on modification (M) is granted, instead a *non-modification* authorisation requirement is specified. The *Amadeus flight service* might need to modify *Flight tickets* to accommodate changes in the itinerary of the *Tourist*, so either the *Tourist* decides he/she does not want the flight tickets modified, or he/she should grant the permission to the *Amadeus flight service*.

4 Conclusions, ongoing and future work

STS-ml is a framework to model and reason over security requirements for STSs. The current version of the framework is a result of an iterative approach, of various evaluations conducted with industrial partners of the FP7 European project Aniketos. The framework has proven suitable to model real case studies spanning from eGovernment to Air Traffic Management Control. STS-ml supports a rich set of security requirements, for which we can identify conflicts.

Ongoing work includes different directions: (i) developing a new version of the tool with improved usability, and (ii) exploring how STS-ml can inform later phases in the design of secure STSs, e.g., the definition of access control policies.

Future work includes: (i) devising other reasoning techniques for more sophisticated reasoning, (ii) exploring techniques for conflict resolution, and (iii) reducing the learning curve for STS-ml, also via self-learning mechanisms.

Acknowledgments. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grants no 257930 (Aniketos) and 256980 (NESSoS).

References

1. F. Dalpiaz, P. Giorgini, and J. Mylopoulos. Adaptive Socio-Technical Systems: a Requirements-driven Approach. *Requirements Engineering*, 18(1):1–24, 2013.
2. F. Dalpiaz, E. Paja, and P. Giorgini. Security Requirements Engineering via Commitments. In *Proceedings of the First Workshop on Socio-Technical Aspects in Security and Trust (STAST)*, pages 1–8, 2011.
3. E. Paja, F. Dalpiaz, and P. Giorgini. Identifying Conflicts in Security Requirements with STS-ml. TR DISI-12-041, University of Trento, `http://disi.unitn.it/~paja/tr-identifying-sec-conflicts.pdf`, 2012.
4. M. P. Singh. An Ontology for Commitments in Multiagent Systems: Toward a Unification of Normative Concepts. *Artificial Intelligence and Law*, 7(1):97–113, 1999.
5. S. Trösterer, E. Beck, F. Dalpiaz, E. Paja, P. Giorgini, and M. Tscheligi. Formative User-Centered Evaluation of Security Modeling: Results from a Case Study. *International Journal of Secure Software Engineering*, 3(1):1–19, 2012.

Analysing Information Integrity Requirements in Safety Critical Systems

Mohamad Gharib and Paolo Giorgini

University of Trento - DISI, 38123, Povo, Trento, Italy
{gharib,paolo.giorgini}@disi.unitn.it

Abstract. Organizations' assets are subject to different threats; which are addressed, usually, by different security solutions. Nonetheless *i** modeling language was not developed with security in mind, which motivated the development of other languages (e.g., SI*) that focused on capturing the security requirements (e.g., privacy) of the system-to-be, but far less attention has been paid for capturing information integrity requirements. Capturing information integrity requirements represents an important need for safety critical systems, where depending on incorrect or inconsistent information may lead to disasters and loss of humans' lives (e.g., Air Traffic Control Management Systems). In this paper we present a novel methodology for developing safety critical systems that extends *i*/* SI* modeling languages with the required concepts and primitives for modeling and analyzing the requirements of safety critical systems, with a special emphasis on information integrity requirements.

1 Introduction

Organizations' assets, especially information, are subject to different kinds of threats. Usually, these threats are captured, prevented or mitigated by different security solutions. The last few years have seen growing efforts for integrating security into the early system development process, since it is the best way to deal with the organizational requirements. For instance, SI* [3] offers a conceptual framework for modeling and analyzing security requirements (mainly privacy and confidentiality) starting from the organizational setting of the system-to-be.

Nowadays, we are more and more experiencing complex systems that are not simply composed by technical components but where organizations, people, and processes become integral part of the system itself. Considering only the technical aspects of the system leaves human, social, and organizational aspects outside the system's boundary and then opening to vulnerabilities that may arise at business and organizational level. For example, in an ATM system, the *ground controller*, based on its role, is not allowed to issue any taxiing information concerning active runways, unless he was permitted by the *local controller*. This can be captured only by the analyzing the system organizational aspects. Furthermore, *Air Traffic Controller Officer (ATCOs)* depends on the *captain* to report the airplane's position when the airplane is out of the radar coverage

area, i.e., *ATCOs trusts* that the *captain* will keep updating (modifying) the location information as required, which prevents inconsistent information within the system that might lead to disasters. Similarly, *trust* can be only captured by the social aspects of the system.

To this end, we advocate that any solution for information integrity related problems should consider the social, organizational and the technical aspects of the system. In this work, we propose a novel methodology for developing safety critical systems that extends *i*/* SI* modeling languages [2,3] with the required concepts and primitives to capture the requirements of safety critical systems with a special emphasis on information integrity requirements. The rest of the paper describes the research objectives, scientific contribution, and finally we outline our conclusions, and discuss the ongoing and future work.

2 Objectives of the research

The main objective of this research is to provide a requirements engineering methodology for developing safety critical systems. In particular, it will provide the following contributions: 1- a modeling language for designing safety critical systems that extends *i*/* SI* modeling languages with the required concepts and constructs for capturing the requirements of such system (especially information integrity requirements); 2- a requirements engineering methodology, that allows for the systematic design of safety critical systems, it aim to support all activities related to requirements analysis. 3- a formal framework to support designers in the requirements verification and validation; and 4- CASE tool to assist designers during the system development process.

3 Scientific contributions

We introduce the new concepts in section (3.1), and the methodology in (3.2).

3.1 The extended modeling concepts

Our modeling language extends the *i*/* SI* modeling languages with several concepts to capture information integrity requirements, including critical information, critical goal, information producer and consumer, and information integrity provision. The first is used to determine which information is critical to the system performance and its integrity has to be preserved, while the second is used to represent the stakeholders' critical objectives, and it is used to determine where information integrity requirements are needed. The third is used to define the initial sources of information and the actor who has full modification permission control over information it produces, while information consumer is used to determine if the integrity of information has been preserved at its final destination. The last is used to represent information provision that is able to preserve the integrity of the provided information. Furthermore, we refine the notions of delegation by introducing the delegation degree based on the trust degree. In the following sections, we define each of these concepts.

Information and information integrity Information can be produced by *information producers* (represent the initial source of information) in several different ways. For instance, information can be *generated* internally or *acquired* from physical objects (e.g., *ATCOs* is able to *acquire* information about airplane location by depending on the radar). Moreover, we call the actor(s) who consume(s) information as *information consumer(s)*, i.e., information is within the objectives of information consumer(s). In the case of safety critical systems depending on incorrect or inconsistent information to perform some activities is not acceptable since it might produce disaster. Thus, preserving information integrity (information integrity requirements) represents an important need for such systems. In this work, information integrity can be evaluated by 3 dimensions, namely, accuracy, completeness and consistency. While information integrity requirements mean preserving these 3 dimensions.

Critical information and critical goals It is well known that not all goals have the same criticality to the organization's performance. Thus, we introduce the *critical goal* concept, which is used to represent the stakeholders' critical objectives, and can be described as any goal that its failure might results in major problems to the organization, i.e., such goals should never fail.

Currently, we define two reasons that might threaten the satisfaction of critical goal: 1- consuming incorrect or inconsistent information; 2- problems that may rise and negatively effects the satisfaction of the goal. The first reason can be avoided by preserving the integrity of information consumed by the critical goal. To this end, we call such information as *critical information* that its integrity should be preserved at any given time. While the second reason can be avoided if all problems that might arise and negatively affect the critical goal satisfaction were detected [1] and solved before its occurrence. For example, *ATCOs* should "manage the airplanes traffic safely" (critical goal) that is why the integrity of information consumed by this goal should be preserved. Furthermore, if *ATCOs detects* that there will be an air traffic increase in his sector, and "manage the air traffic safely" is threatened, he should take some actions to *solve* this *problem* by avoiding its occurrence (e.g., delay or change the path of some flights).

Delegation and trust An actor might not have the capabilities to fulfill his objectives (goals). Thus, it delegates them to other actors. SI* introduces the notion of goal delegation, which identifies the transfer of responsibilities concerning a goal satisfaction among actors, where an actor (delegator) delegates the achievement of a goal (delegatum) to another actor (delegatee). In this work, we proposed a refinement of goal delegation by introducing the notions of delegation and trust. Moreover, we introduce 4 different degrees of trust [full, partial, limited, no] trust. Consider for example, an *ATCOs* delegates the goal "manage safely separation with other planes" to the *airplane captain*, the trust between *ATCOs* and the airplane captain concerning the goal satisfaction, will be eval-

[1] Certain information is used to detect the expected occurrence of each problem

uated based on the *airplane captain's* capabilities, problems detecting [2] and solving.

Full trust: *ATCOs* has a full trust that the *airplane captain* is able to satisfy the goal "manage safely separation with other planes", if and only if, the captain is able to satisfy the goal, and he is able to detect and solve all problems that may rise during the goal satisfaction.

Partial trust: *ATCOs* has a partial trust that the *airplane captain* is able to satisfy the goal "manage safely separation with other planes", if the captain is able to satisfy the goal, but he is not able to detect all the problems that may rise during the goal satisfaction, even he is able to solve them.

Limited trust: *ATCOs* has a limited trust that the *airplane captain* is able to satisfy the goal "manage safely separation with other planes", if the captain is able to satisfy the goal, but he is neither able to detect nor solve all the problems that may rise during the goal satisfaction.

No trust: *ATCOs* has no trust that the *airplane captain* is able to satisfy the goal "manage safely separation with other planes", if, simply, the captain is not able to satisfy the goal.

Furthermore, we extend the notion of goal delegation introduced in the previous languages (e.g., SI*) based on the different degrees of trust. We introduce three types of goal delegation: 1- full delegation; 2- partial delegation; and 3- limited delegation, they can be defined as follows:

Full delegation: The delegator fully trusts that the delegated goal will be satisfied, since the delegatee is able to satisfy the goal, and it is able to detect and solve any problem that may arise during the goal satisfaction.

Partial delegation: The delegator partially trusts that the delegated goal will be satisfied, since the delegatee is able to satisfy the goal, but it is not able to detect all problems that might rise during the goal satisfaction even if it is able to solve them.

Limited delegation: The delegator limitedly trusts that the delegated goal will be satisfied, since the delegatee is able to satisfy the goal, but it is neither able to detect nor solve problems that might rise during the goal satisfaction.

Ex1. *ATCOs fully delegates* "manage safely separation with other planes" to *captains*, if they were flying under Visual flight rules (VFR), where VFR require a *captain* to be able to control the airplane's attitude, navigate, and avoid obstacles by itself, i.e., they are able to detect and solve all the problems that may rise during the satisfaction of the goal.

Ex2. *ATCOs partially delegates* "manage safely separation with other planes" to *captains* at airways intersections, since *captains* are not able to detect if the intersection is being used by others, even they are able to solve such problem.

[2] The integrity of information used to detect the expected occurrence of each problem has to be preserved, especially in the case of goal delegation, since it is not necessarily that the actor who detect the problem is the same actor who supposed to solve it

Ex3. *ATCOs limitedly delegate* "airplane safely landing" to *airplane captains*, since *captains* are not able to neither detect nor solve any unexpected problem that might rise during the satisfaction of this goal.

In this case of partial and limited delegation the goal satisfaction is threatened, since the delegator does not have a full trust that the delegatee can satisfy the goal. It is well known that the lack of trust can be tolerated with monitoring the delegatee performance. In this work, we introduce two types of monitoring:

Partial monitoring: the monitoring actor knows all the possible ways in which the achievement of the goal can be performed by the monitored actor, i.e., the monitoring actor is able to detect if the monitored actor is not performing the achievement of the goal by one of these ways.

Full monitoring: the monitoring actor knows exactly how the achievement of the goal should be performed by the monitored actor, i.e., the monitoring actor is able to detect if the monitored actor is not performing the achievement of the goal as planned at any moment.

Ex4. In **Ex2** *ATCOs* should *partially monitor* the delegated goal satisfaction, since only the *ATCOs* is able to *detect* such problems, and asks the *captain* to alter his flight path when it is needed.

Ex5. In **Ex3** the *ATCOs* should *fully monitor* the delegated goal satisfaction, since *captain* is not able to neither detect nor solve any unexpected problems that may rise during the satisfaction of this goal.

3.2 The requirements engineering methodology

The methodology aims for the systematic design of safety critical systems, it is intended to support all activities related to requirements analysis process, including the requirements verification and validation process to determine whether the model satisfies the stakeholders' requirements, and to ensure that the requirements are correct, complete, and consistent [3]. The process starts with the actor modeling, in which actors are modeled along with their objectives, entitlements, and capabilities. Then, critical goals are defined, goals are analyzed and refined, and the criticality is propagated in the case that the critical goal. Based on the criticality of the consuming goal critical information is determined. Goal delegation, information [integrity] provision are modeled. Furthermore, social modeling starts with trust modeling, and based on the trust degree the delegation types are determined and modeled. Finally, based on the goal delegation type, full or partial monitoring are added. The methodology, including the new concepts, notations and the tools, will be evaluated using an ATM case study. Figure 1 shows the main phases of the requirements analysis process.

[3] Not included in this paper, some of the concepts are formalized in [1]

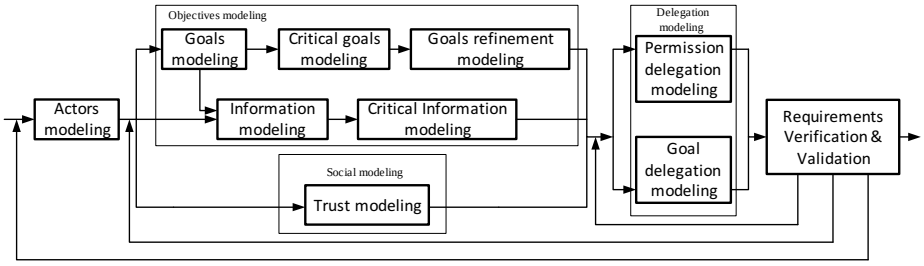


Fig. 1. Requirements Analysis Process

4 Conclusions

In this paper, we extended *i */* SI* with several concepts for capturing the requirements of safety critical systems, we focus more on information integrity requirements, and refine the notions of goal delegation based on the different degrees of trust. Furthermore, we showed how the requirements of the system-to-be will be constructed by the methodology.

5 Ongoing and future work

We are considering the following topics for the future work:

- The modeling language will be extended for capturing the related information integrity dimensions (accuracy, completeness and consistency).
- The methodology will be extended to capture the requirements of business critical systems beside the safety critical systems.
- We intend to increase the number of the design properties that our model is able to check (currently we have 8).
- A CASE-Tool that allows designers to verify the correctness of the model will be developed.

Acknowledgments The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant no 257930 (Aniketos) and 256980 (NESSoS).

References

1. Mohamad Gharib and Paolo Giorgini. Analysing information integrity requirements in safety critical systems. *The 3rd International Workshop on Information Systems Security Engineering WISSE13. To appear.*, 2013.
2. Eric Siu-Kwong Yu. *Modelling strategic relationships for process reengineering.* Ph.d. thesis, Toronto, Ont., Canada, Canada, 1996. AAINN02887.
3. N. Zannone. *A requirements engineering methodology for trust, security, and privacy.* PhD thesis, PhD thesis, University of Trento, 2006.

From Requirements to Architectures for Better Adaptive Software Systems

João Pimentel[1,2], Konstantinos Angelopoulos[2], Vítor E. Silva Souza[3], John Mylopoulos[2], and Jaelson Castro[1]

[1] Centro de Informática, Univ. Federal de Pernambuco (UFPE), Recife, Brazil
{jhcp,jbc}@ufpe.br
[2] Department of Information Eng. and Computer Science, University of Trento, Italy
{angelopoulos,jm}@disi.unitn.it
[3] Computer Science Dept., Federal University of Esprito Santo (Ufes), Vitória, Brazil
vitorsouza@inf.ufes.br

Abstract. The growing interest in developing adaptive systems has led to numerous proposals for approaches aimed at supporting their development. Some approaches define adaptation mechanisms in terms of architectural design, consisting of concepts such as components, connectors and states. Other approaches are requirements-based, thus concerned with goals, tasks, contexts and preferences as concepts in terms of which adaptation is defined. By considering only a problem- or a solution-oriented view, such proposals are limited in specifying adaptive behavior. In this paper we present ongoing work on supporting the design and runtime execution of adaptive software systems both at a requirements and architectural level, as wells as its challenges, ranging from architectural derivation from requirements to refined adaptation control mechanisms.

Keywords: adaptive systems, architectural design, adaptation control mechanisms, requirements

1 Introduction and Objectives

In [1], the authors conducted a comparative study, concluding that requirements- and architecture-based approaches for software adaptation share common elements, such as the use of feedback loops and of external control mechanisms. However, there are also differences that reveal complementary advantages and disadvantages of the two approaches. On one hand, requirements-based approaches capture and model the objectives of the system, but they lack awareness about the capabilities and the limitations of the proposed solution. On the other hand, architectural models provide guidance for the deployment of the monitoring mechanisms and the effectors that apply the adaptation process on the target system. The objectives of the system, however, are coded into the adaptation strategy, making it difficult to handle changes at the requirements level.

Based on the results of this study, the authors have embarked on a research project to better link requirements and architectural models by (a) developing

techniques for deriving architectural models from requirements, and (b) extending existing techniques for designing adaptive software so that they exploit both requirements and architectural models.

2 Baseline

Our baseline is the *Zanshin* framework for the design of adaptive systems [2–4], which in turn is founded on Goal-Oriented Requirements Engineering (GORE) [5]. Adopting from Control Theory the concept of feedback loop for adaptation, *Zanshin* augments goal models with requirements for monitoring and adaptation of such loops.

To illustrate, Fig. 1 shows a goal model for an adaptive Automated Teller Machine (ATM). Traditional $i^\star$ elements (goals and tasks) are connected by refinement/operationalization relations, using AND/OR Boolean semantics for goal satisfaction. *Zanshin* introduces *Awareness Requirements* (*AwReqs*, represented by small circles) and *Control Variables* (rep. by diamonds).

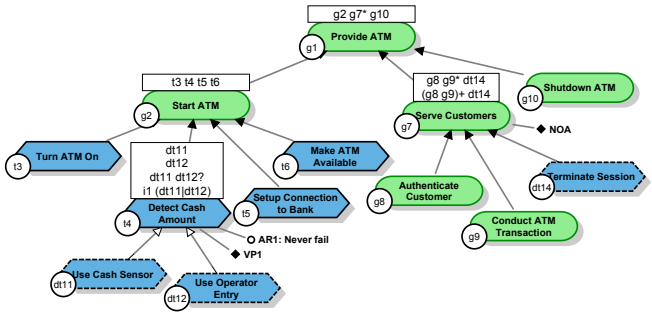


Fig. 1. Goal-based requirements specification for an ATM.

AwReqs are the requirements for the monitoring component of the feedback loop and impose requirements on the success/failure of other requirements by talking about the states assumed by other requirements at runtime [2]. As such, they represent situations in which the stakeholders would like the system to adapt. In Fig. 1, *AwReq AR1* states that task *Detect Cash Amount* should never fail.

Control Variables (CVs) are elicited during *System Identification* [3], alongside *Variation Points* (VPs) and qualitative relations between these two parameters (CVs and VPs) and indicators of requirements convergence, namely, *AwReqs*. Examples in Fig. 1 are the VP for *Detect Cash Amount* (two ways of satisfying it) and the CV *Number of Operators Available* (*NOA*). Differential relations, e.g., $\Delta(AR1/NOA) > 0$ indicate how changes in parameters affect indicators (in this case, increasing *NOA* increases the success of *AR1*).

Lastly, *Evolution Requirements* (*EvoReqs*) [4] specify when and how should other requirements change at runtime. For example, an *EvoReq* may be "If requirement R fails three times in a row, replace it with requirement $R-$", where $R-$ is a weaker (i.e., easier to fulfil) requirement. Using these new classes of requirements, *Zanshin*[1] implements a feedback loop that supports adaptation of a base system.

3 Research Agenda

We are interested in taking further the baseline by combining requirement models with software architectures. Towards this end, we propose a systematic methodology for deriving architectural models from requirements. This research will allow us to design adaptive software systems that exploit combined goal and architectural models, thereby capturing allowable adaptations at both levels of abstraction. Therefore, the system would have the maximum variety of alternatives when it has to deal with failures or with environmental changes. The second part of this work involves extending *Zanshin* to exploit combined goal and architectural models, but also making it quantitative, in order to acquire higher precision. Moreover, *Zanshin* will be extended to deal with multiple failures, exploiting techniques inspired by Control Theory.

3.1 Architectural derivation

Architectural derivation is concerned with the generation of architectural models, which can include: (a) components & connectors models for describing the system structure; (b) statecharts for describing system behavior; and (c) feature model for expressing the variability of system configuration. These different models are complementary, each one capturing a particular view of the system being designed, thus requiring different derivation approaches.

In previous work [6] [7], we proposed methods to derive the aforementioned models from goal models. The key of that proposal was to derive the models in such a way as to preserve the variability expressed in the goal model. However, when considering architectural derivation and its design decisions for the particular case of adaptive systems, there are three new concerns that arise:

a. *Additional variability* — there may be different alternatives to accomplish a given task. For instance, different algorithms and different technologies can be applied, each with its different benefits and drawbacks. The alternatives identified during architectural derivation will expand the space of adaptation possibilities.
b. *Additional control elements* — besides referring to requirements concepts, *Zanshin* elements (such as *AwReqs* and *Control Variables*) may also refer to and have an influence on architectural concerns. For instance, the time interval for a timed transition could be defined as a *Control Variable*, rather than as a pre-defined, static interval.

[1] See `https://github.com/sefms-disi-unitn/Zanshin/wiki`

c. *Additional features to support adaptation* — the support of self-adaptation may require the inclusion of new features in the system. This is the case, for instance, when the system requires some kind of instrumentation in order to monitor the satisfaction of *AwReqs*.

In [8] we handled the identification of additional features, considering the monitoring capabilities required to monitor runtime context. There, we were concerned with the derivation of components & connectors. An approach for eliciting future requirements, which can be used to identify additional variability (both at requirements and architectural level) was presented in [9]. In [10] [11] we explore additional variability derived from different web services that are available in a pool of services.

Currently, we are working on including additional variability and additional control elements, while supporting the derivation of statecharts. After all, statecharts capturing system behavior constitute the most important architectural view for adaptive systems.

The process for deriving statecharts from goal models comprises 7 steps. The first step, *Identify design tasks and constraints*, allows to refine the requirements model by including elements that are relevant from the architectural point of view. Next, *Assign tasks*, consists of assigning the tasks that will not be performed nor supported by the base software system — e.g., tasks that will be performed by an external actor (human or otherwise). In the next step, *Define basic flow*, the architect analyzes all refinements of the goal model and defines flow expressions that define their runtime behavior. These expressions, which allow to define flows with a notation akin to regular expressions, are used in the next step (*Generate base statechart*) to create a skeleton of the statechart. The statechart depicted in Fig. 2 was derived from the goal model in Fig. 1. For this statechart, we selected the third flow expression of *Detect Cash Amount (dt11 dt12?* — perform dt11, then optionally perform dt12) and the first flow expression of *Serve Customers (g8 g9* dt14* — perform g8, then perform g9 zero-or-more times, lastly perform dt14).

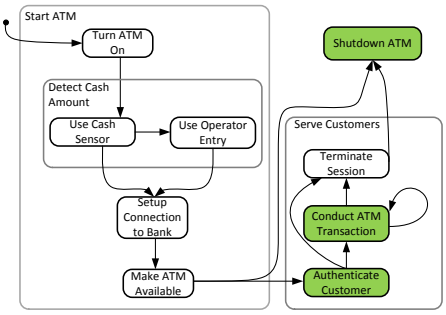


Fig. 2. Statechart for a partial behavior refinement of the ATM system

In the remaining steps the statechart skeleton is refined, as follows. First, during *Specify transitions* the architect defines events and conditions of the derived transitions. Then, the statechart is enriched to describe the system's adaptive behavior, including the interaction with an external component that provides adaptation-related functionality. This takes place during *Specify adaptive behavior*. As a last step, *Perform further refinements* allows the architect to expand the model in order to include technical details and other concerns that may not have been handled earlier, by exploiting the statechart concept of sub-states.

3.2 Advancing runtime software adaptation mechanisms

Section 2 sketched *Zanshin* and how it defines adaptation mechanisms based on parameters and indicators as well as qualitative relations among them. Given the subjective nature of requirements, it is important to support such qualitative mechanisms. However, in some scenarios, it is possible to identify quantitative relations instead of qualitative. This is especially true for architectural models, which are more tangible than requirements ones.

The use of quantitative relations will allow finer tuning of parameters when restoring failed indicators, which in turn can reduce critical overshooting and redundant oscillations. Methods such as regression analysis can then be used to extract quantitative information about the relation among parameters and indicators.

Moreover, such quantitative relations assist in solving the issue of multiple failing indicators. Usually, software systems involve conflicting requirements (e.g., cost and performance) that result in conflicting indicators (i.e., when one is failing the other is succeeding and vice versa). When several indicators fail it is hard to perform a trade-off analysis with a good degree of confidence using only qualitative information. Thus, the quantitative relations, combined with prioritized indicators, can be exploited in order to apply optimization techniques and identify the best possible adaptation.

4 Conclusions

We have presented ongoing work towards improving support for the development of adaptive software systems. On one hand, the combination of requirements and architectural models will provide a richer space of possible adaptations. The proposed derivation methodology will facilitate the creation of adaptive systems based on the *Zanshin* framework. On the other hand, the control mechanisms of the framework itself will be improved, by tackling the occurrence of multiple (and possibly conflicting) failures and using quantitative relations to increase the precision of adaptation mechanisms.

As mentioned in Section 2, a prototype implementation of the *Zanshin* framework is available. This implementation will be extended in order to support the

enhancements proposed in Section 3.2. A prototype tool for the derivation of statecharts, as presented in Section 3.1, is currently under development[2].

Acknowledgments. This work has been supported by the ERC advanced grant 267856 "Lucretius: Foundations for Software Evolution" and by Brazilian institutions CAPES and CNPq.

References

1. K. Angelopoulos, V. E. S. Souza, and J. Pimentel, "Requirements and Architectural Approaches to Adaptive Software Systems: A Comparative Study," in *Proc. of the 8th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (to appear)*, 2013.
2. V. E. S. Souza, A. Lapouchnian, W. N. Robinson, and J. Mylopoulos, "Awareness Requirements," in *Software Engineering for Self-Adaptive Systems II* (R. Lemos, H. Giese, H. A. Müller, and M. Shaw, eds.), vol. 7475 of *Lecture Notes in Computer Science*, pp. 133–161, Springer, 2013.
3. V. E. S. Souza, A. Lapouchnian, and J. Mylopoulos, "System Identification for Adaptive Software Systems: A Requirements Engineering Perspective," in *Conceptual Modeling ER 2011*, pp. 346–361, 2011.
4. V. E. S. Souza, A. Lapouchnian, K. Angelopoulos, and J. Mylopoulos, "Requirements-driven software evolution," *Computer Science - Research and Development*, pp. 1–19, 2012.
5. J. Mylopoulos, L. Chung, and E. S. K. Yu, "From Object-Oriented to Goal-Oriented Requirements Analysis," *Communications of the ACM*, vol. 42, no. 1, pp. 31–37, 1999.
6. Y. Yu, J. C. S. do Prado Leite, A. Lapouchnian, and J. Mylopoulos, "Configuring features with stakeholder goals," in *Proceedings of the 2008 ACM symposium on Applied computing - SAC '08*, pp. 645–649, ACM Press, 2008.
7. Y. Yu, A. Lapouchnian, S. Liaskos, J. Mylopoulos, and J. C. S. P. Leite, "From Goals to High-Variability Software Design," in *Foundations of Intelligent Systems*, vol. 4994/2008, pp. 1–16, 2008.
8. J. Pimentel, M. Lucena, J. Castro, C. Silva, E. Santos, and F. Alencar, "Deriving software architectural models from requirements models for adaptive systems: the STREAM-A approach," *Requirements Engineering*, vol. 17, no. 4, pp. 259–281, 2012.
9. J. Pimentel, J. Castro, H. Perrelli, E. Santos, and X. Franch, "Towards anticipating requirements changes through studies of the future," in *5th International Conference on Research Challenges in Information Science*, pp. 1–11, IEEE, 2011.
10. J. Pimentel, J. Castro, E. Santos, and A. Finkelstein, "Towards Requirements and Architecture Co-evolution," in *Advanced Information Systems Engineering Workshops*, pp. 159–170, 2012.
11. X. Franch, P. Grunbacher, M. Oriol, B. Burgstaller, D. Dhungana, L. Lopez, J. Marco, and J. Pimentel, "Goal-Driven Adaptation of Service-Based Systems from Runtime Monitoring Data," in *2011 IEEE 35th Annual Computer Software and Applications Conference Workshops*, pp. 458–463, IEEE, July 2011.

[2] Available at `https://github.com/jhcp/GoalArch/tree/master/papers`

Using i* to Capture Consumer Preferences as Requirements for Software Product Lines

Jelena Zdravkovic, Constantinos Giannoulis, Eric-Oluf Svee

Department of Computer and Systems Sciences
Stockholm University, Forum 100, SE-164 40 Kista, Sweden
{jelenaz, constantinos, eric-sve}@dsv.su.se

Abstract. The need for software to fit to diversity of numerous consumers has become a norm. Furthermore, technology innovations stimulate the growth of such software, thus making it even more available and appealing to consumers. Although how economic values relate and influence IT systems is an area that has been addressed, it is not clear whether and how consumer values do so. To address this challenge, this study aims to using i* establish a link between preferences of consumers and system requirements for Software Product Line (SPL) as a seamless way for systematically realizing variations. The presented results are grounded in an empirical study related to the development of a system for Online Education.

Key words. Consumer Value, SPL, Goal Modeling, i*, System Requirements

1 Introduction

There are a number of types of value: quantitative, or economic, are generally understood as an amount in goods, products, services or money, considered as a suitable equivalent for something else: a fair price or return for an investment [1]. In contra poise are values with a qualitative nature, detailing how a good, product, or service is delivered to, or perceived by, the consumer. These have been variously termed non-economic values, internal values, and most often - *consumer values* [2].

While the impact of quantitative values on information systems is readily seen and acknowledged, particularly within software engineering, qualitative values have been researched to a much lesser degree, in particular consumer values. Several attempts to address this deficiency in non-economic values within the development space have been made, however, none have had an explicit consumer focus. This is a failing because the values of an individual have an effect on their behavior as consumers [3].

Take, for instance, a situation where Consumers A and B both want “convenient” book delivery. For A that means downloading an electronic book immediately, while for B it means dispatching the book via post quickly to a location near to the consumer’s residence. The consumer value “convenient” is not clearly defined, so the business cannot develop the proper support systems required to deliver what the consumers’ desire. As both a theory and a set of practices Software Product Lines

(SPL) promises to address the challenges outlined above through the design of software products sharing a common set of features, which at the same time are specialized to satisfy the specific needs of a particular market segment.

2 Objectives of the Research

The objective of this study is to present how consumer preferences can be captured in the development of IT systems, by using the theory for the design of a collection of similar software products, namely SPL. We propose a method for linking consumer values with Goal-Oriented Requirements Engineering (GORE) approaches, as they are acknowledged for effective exploration of alternatives in requirements, and more specifically, for elicitation of variable and common requirements of SPL. Using the i* framework as the example for GORE, we also leverage from the existing proposals the ability to link goals to feature models, leading further to the configuration of SPL.

The research approach taken in this paper is conceptual and empirical. Concepts used in value modeling and consumer representation are integrated with those of i* and SPL. The theory is tested through a study on Online Education systems.

3 Scientific Contributions

3.1 Consumer Values

Holbrook's Typology of Consumer Value [2] classifies the preference of individuals concerning the goods or services that they evaluate for a potential use, or appraise from their previous consummations. According to Holbrook, a consumer value is a) *interactive*, as it entails an interaction between a subject and an object, b) *relativistic* refers to consumer values being comparative, c) *preferential* as consumer values are the outcome of an evaluative judgment, d) and *experience* meaning that consumer values not reside in the product/service acquired, but in the consumption experience.

Three consumer value dimensions are the basis of Holbrook's typology: *Extrinsic/Intrinsic*, *Self-oriented/Other-oriented*, and *Active/Reactive*. Based on them, eight archetypes representing distinct types of value in the consumption experience are derived (Table 1):

Table 1. Holbrook's Typology of Consumer Values

Extrinsic	*Intrinsic*		
Efficiency	**Play**	*Self-Oriented*	*Active*
Excellence	**Aesthetics**		*Reactive*
Status	**Ethics**	*Other-Oriented*	*Active*
Esteem	**Spirituality**		*Reactive*

Empirical Study. The main challenges in online education concern creating software for courseware appealing to diverse students. To encourage students' attention and

learning, one of the crucial factors is to design software systems in the way to support both intrinsic and extrinsic motivations/values of students.

In our research we have performed an empirical study with the students of Stockholm University, Non-Master and Master, to examine their preferences for the online education system, as well as to assess their importance. For the first, we have profiled Consumer Values through individual interviews, where the students were asked to describe their preferences for an online education system in terms of each of the eight Holbrook's value archetypes. From around 220 value examples that were collected, in Table 2 below we present some of them.

Table 2. Examples of Consumer Values obtained during the interview process.

Holbrook Value	Example (Measure)
Ethics	-Prevent cheating -Provide materials -Communication rules -Promote professionalism
Play	-Discussion with others -Whimsical -Provide fun learning -Make layout customizable -Provide push-pull functions
Aesthetics	-Access (through web browser or app) -Interactive -No mountains of text
Efficiency	-Save time -Access whenever/wherever -Time limits for completing assignments

To assess the importance of 8 archetype values (Table 1), we have used the survey instrument of the Basic Value framework of Schwartz [2], administered as the European Social Survey (ESS) - we have collected the weights (importance) of generic values of a large sample of students (>200), and owing to the established mapping between the value frameworks of Schwartz and Holbrook [4], we assessed the importance of the concretizations of Consumer Values of Holbrook elicited in the individual interviews (Table 2) according to the survey's results. As an illustration, the survey results showed that Ethics is the most important to Non-Master students, while Play to Master students. Another finding was that Aesthetics and Efficiency are highly weighted from the both student segments, though in the opposite order.

3.2 From Consumer Preferences to Requirements for SPL

Once a product is examined by different consumer segments for desired properties and their importance, the collected information can be transformed further to a requirements model, with the purpose to configure a Software Product Line. Based on

[5], where the mappings between e^3 value model and i* goal framework are analyzed, we propose the construction of an i* SR model from Holbrook's consumer framework by mapping:

- Consumer (Student) and product Provider (University), to distinct *Actors* in i*.
- The exchange of the economic value (Online Education), to *Resource Dependency* in i*, further elaborated within Provider, through a System type sub-Actor.
- Consumer Value to the *Beliefs* of the consumer i* actor, i.e. to the conditions about the world that the consumer holds to be true; the beliefs become the *Soft Goals* of the Provider, i.e. the intentions without a clear-cut criterion of achievement, thus requiring further refinement (decomposition).
- The weight of a population's consumer value, to a numbered annotation* in the belief representing the value (ex. Play has priority 3 for non-master, and 1 master)

* Another way to manage priorities is based on the notion of precedence of goals, and among goals [6]; however, since there is no i* implementation available, we have not considered this extension in our study.

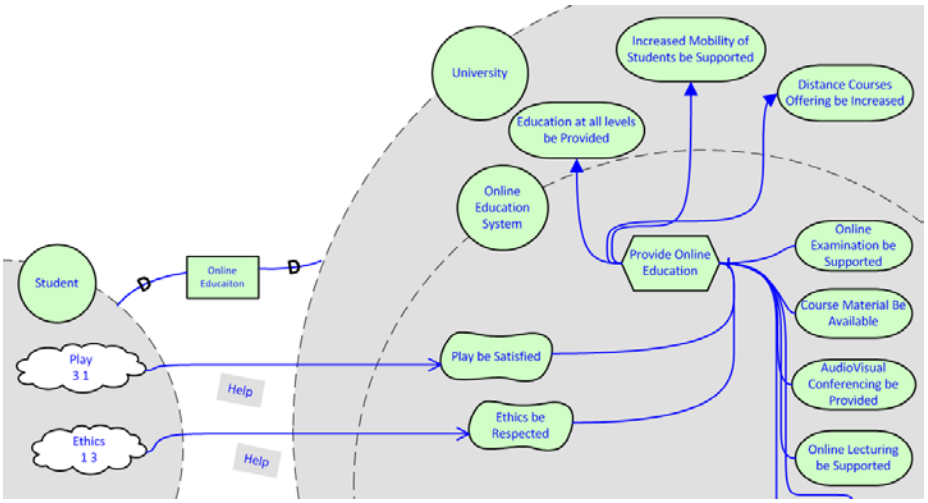


Fig. 1. An excerpt of the top-level i* model for the Online Education System product line. Apart from the elements derived from the mappings from the Consumer Value framework, the model also captures the core goals of the University, and of the Online Education System.

The refinements of Holbrook's consumer values (see Table2 for examples), concretizing (i.e. measuring) the eight archetype values from Table 1 for a software product in the consideration, are modeled as Resources or Tasks in i* through the decompositions of the soft goals corresponding to the values. In the following figure we present a detailed part of the i* SR model where Ethics, the value highly assessed by the both student sub-populations (priority 1 for non-master, and 3 for master) is further elaborated:

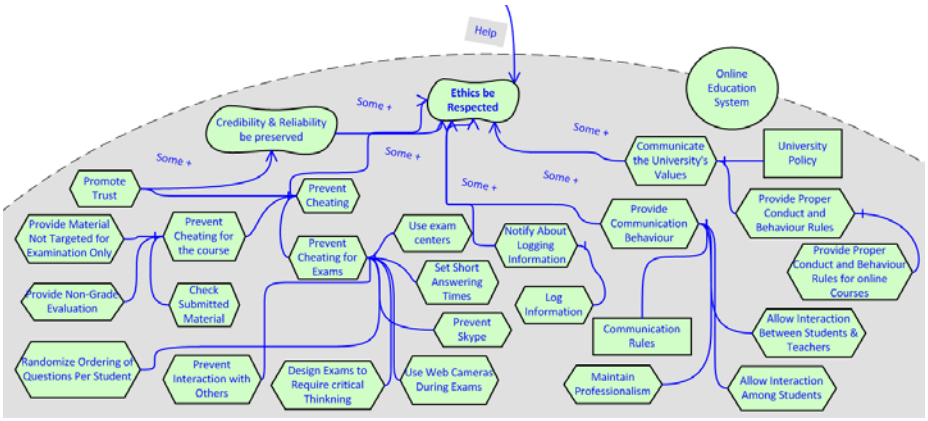


Fig. 2. Decomposition of Soft Goal "Ethics be Respected".

The "system" (Online Education System in Figure 2) is the actor representing future SPL. To link the i* SR model with a configuration for SPL, the theory of feature modeling is applied, where features are used as the basis for analyzing and representing *commonality* and *variability* of systems in a solution domain [7].

For the elaboration of the requirements for SPL from an i* SR model using features, the guidelines concerning feature identification presented in a goal oriented approach for SPL (G2SPL) [8] are considered. The main difference is that in our approach the features originate from consumer values, and as such, in addition to their elicitation, they are classified (as common or variable) and prioritized in an early stage of the requirements collection - i.e. within i*, according to the mappings from consumer preferences. A small sub-set of the feature elicited in the empirical study for the value Ethics is presented in Table 3 below. In the table, common features are the ones identified in both segments. For example, *Prevent Cheating for Exams* is a feature elicited as a preference by the both student segments. Variable features are the ones not identified in both segments, i.e. including a priority number for some segments and "-" for at least one segment. For example, *Provide Communication Rules* is a feature elicited only by Non-Master students.

Table 3. Excerpt with features for the soft-goal "Ethics be Respected".

Element	FatherElement	Feature	Priority	
			NonM	Master
Prevent cheating for exams	Prevent cheating	Prevent cheating for exams	**1**	**3**
Log information	Notify about logging	Log all events and documents	**-**	**3**
Provide communication behavior	-	Provide communication behavior	**1**	**3**
Communication rules	Provide communication behavior	Provide Communication rules	**1**	**-**

The features identified valid for Master students will be the subjects of the requirements for the online education system product developed for them, while the features identified to be valid for Non-Master students will be part of the other product. For more details about mapping the features to requirements, see [8].

4 Conclusions

In this paper we have presented a consumer-based approach to collecting requirements for SPL. The use of the consumer framework for elicitation of requirements for SPL is argued and demonstrated through the mappings of a consumer value framework to the i* goal framework for RE. The objective has been to elevate the alignment between user needs and the final software by proposing a systematic approach for structuring of a diversity of preferences of consumers bundled into a SPL.

5 Ongoing and Future Work

The main ongoing work concerns further development of our Consumer Preference Meta-Model (CPMM) [9], meant to integrate the core elements of the business value modeling, as well as those of consumer frameworks, such as preferences, segmentation, context of use, and preference measures.

References

1. McCarthy, W. E.: The REA accounting model: A generalized framework for accounting systems in a shared data environment. *Accounting Review*, pp. 554–578 (1982)
2. Holbrook, M.: *Consumer value: a framework for analysis and research*. Routledge (1999)
3. Schwartz, S. H., Melech, G., Lehmann, A., Burgess, S., Harris, M., & Owens, V.: Extending the cross-cultural validity of the theory of basic human values with a different method of measurement. Journal of cross-cultural Psychology, 32(5), pp. 519–542 (2001)
4. Svee, E.O., Zdravkovic, J., and Giannoulis, C.: Consumer Value-Aware Enterprise Architecture. In Proceedings of ICSOB'12, LNBIP 114, 55–69 (2012)
5. Raadt van der B., Gordijn, J., Yu, E.: Exploring web services ideas from a business value perspective. In Proceedings of IEEE RE'05, IEEE Computer Society, 53-62 (2005)
6. Liaskos, S., McIlraith, Sh., Sohrabi, Sh., and Mylopoulos, J.: Representing and reasoning about preferences in requirements engineering. Requirements Engineering Journal (REJ). 16(3), pp. 227-249 (2011)
7. Czarnecki, K., Helsen, S., and Eisenecker, U. W.: Staged configuration using feature models. In Proceedings of SPLC'04, 266–283 (2004)
8. Silva, C., Borba, C. and Castro, J.: A Goal Oriented Approach to Identify and Configure Feature Models for Software Product Lines. In: Proc. of the WER'11, Brazil (2011)
9. Svee, E.O., Giannoulis, C., and Zdravkovic, J.: Towards Consumer Preference-Aware Requirements. CAiSE 2012 Workshops (BUSITAL), LNBIP 112, pp. 531–542 (2012)

Model Contextual Variability for Agents Using Goals and Commitments

George Chatzikonstantinou and Kostas Kontogiannis

School of Electrical and Computer Engineering
National Technical University of Athens,Greece
gchatzik@cslab.ece.ntua.gr, kkontog@softlab.ntua.gr

Abstract. Goal models have been extensively utilized in requirements engineering as they provide an expressive and qualitative way to represent requirements, while recent extensions related to contextual variability have further increased the expressiveness of the models. In addition to their application in requirements engineering however, goal models have been also proposed in the literature as a formal way to define the internal design of agents in multi-agent systems. In this paper we adopt the idea of applying goal models with contextual elements, i.e. conditional goals, decompositions and contributions, as a mean to model the internal design of agents. Furthermore, we express the way those agents can interact with each other in terms of commitments, a recently introduced modeling concept that can be used for the definition of communication protocols in multi-agent systems. In this context, we introduce a transformation process that maps all conditional elements to commitments and contributions, and hence, reasoning techniques that exist for commitments can be applied to contextual models with no further changes.

Keywords: goal models, agent-oriented models, commitments

1 Introduction

Goal models, which have been extensively utilized in requirements engineering, have been proposed in the literature as a formal way to define the internal design and objectives of agents in multi-agent systems. This idea has been also adopted by Chopra et al. in [1], where they additionally introduce the notion of *commitment* as a way to model the communication protocol, i.e. the way agents can interact with each other, in multi-agent systems. More specifically, in [1], the internal design of each agent, along with the goals it wants to achieve, are defined in terms of AND/OR decompositions, and contributions. Given that an agent has the capability to fulfil only a subset of its intended goals on its own, it cannot help but depend on others for the fulfilment of the remaining ones. These dependencies are expressed via roles the agents can adopt, and commitments that exist between agents that play those roles. In a more detailed manner, a commitment of the form *Commitment*(*Debtor*,*Creditor*, *antecedent*, *consequent*) means that the Debtor is committed to the Creditor for the consequent if the

2

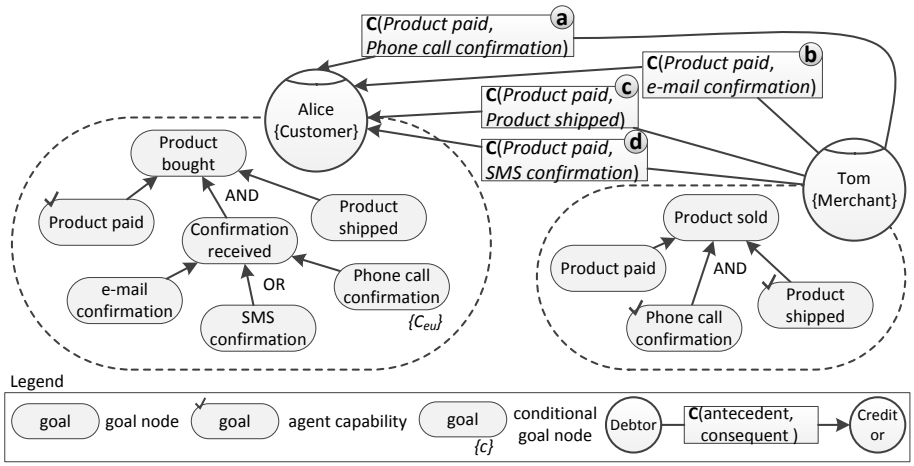


Fig. 1: Two agents that participate in an online marketplace application.

antecedent holds, where both Debtor and Creditor are roles that an agent can adopt. Once the goal model of each agent, as well as the sets of roles and commitments have been specified, the authors provide the semantics of a reasoning process that can be used to check whether the achievement of a specific agent's goal can be supported by the underlying protocol.

As an example consider the case illustrated in Fig. 1, where two agents, namely Alice and Tom, participate in an online marketplace application (ignore the conditional goal node for the moment). In this example Alice, who adopts role "Customer", can satisfy her goal "Product paid" on her own as this is a *capability* of hers. The satisfaction of her goal "Product shipped" however, can be only supported by the commitment denoted as "c" in the figure. In a nutshell, because of the existence of the commitment "c", Tom, who participates in the application as a "Merchant", is committed to provide "Product shipped", if Alice (the "Customer") fulfils "Product paid".

2 Research Objectives

In this paper, we utilize the ideas of [2] and [3] related to contextual variability of goal models, in order to capture the variability that may exist in the internal design of an agent as a consequence of alterations in the context an agent acts in. In this case however, the problem that needs to be solved is slightly different. We are now interested in studying whether a specific agent's goal can be supported by the underlying protocol *and within the given context*. Hence, the reasoning process described in [1] does not apply any more.

To overcome this problem, and in an attempt to keep the reasoning mechanism unchanged, we propose the framework illustrated in Fig. 2. In a more descriptive manner, we firstly extend the metamodel introduced in [4] so as to

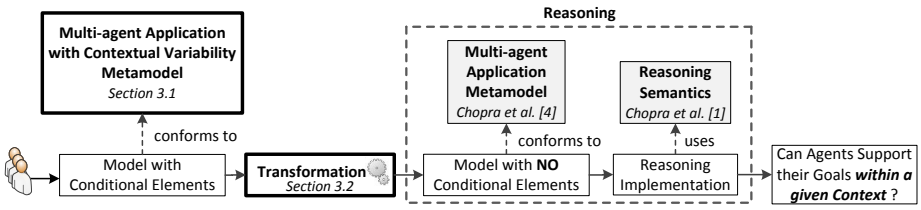


Fig. 2: The proposed framework.

support contextual elements, i.e. goals, decompositions and contributions (see section 3.1), and we then introduce a transformation process that will allow us to express those context-related modeling elements in terms of commitments and contributions (see section 3.2). Thus, by applying this transformation, we can produce a model that captures the multiple variations that may exist in the initial model, while at the same time we can reuse the reasoning semantics proposed in [1]. It is important to note that the produced model is only intended to be used for the generation of the rules required for the reasoning process.

3 Scientific Contribution

3.1 Agents with Context-dependent Goal Models

Using the conceptual model defined in [4] as a starting point, and taking into consideration the way contextual variability is captured in goal models in [2] and [3], we ended up with the metamodel depicted in Fig. 3. While the part of the metamodel related to the definition of a service engagement in terms of commitments and roles is identical to the one introduced in [4] (and thus not included in the figure), the goal-oriented representation of agents is extended with the addition of *conditions*. More specifically, the proposed metamodel allows for goals, decompositions, and contributions to be defined as *conditional*, in the sense that each one of those modeling elements can be related to one or more conditions, the truth values of which dictate the existence of the corresponding element in the goal model. Hence, a conditional element is included in the model only if at least one of the attached conditions is true, otherwise it must be removed. In this respect, the metamodel can capture the contextual variability of the internal model of an agent, where a specific context is mirrored by the assignment of truth values to all conditions in the model.

For the example illustrated in Fig. 1, lets assume that the online marketplace application serves only e-shops that are located in Europe, and that the goal node *"Phone call confirmation"* is conditional and exist in the model only if Alice is located in Europe when she interacts with the application (C_{eu} is true), and is removed otherwise. Lets also assume that Alice happens to be in the USA (i.e. condition C_{eu} is false), and hence she can only receive a confirmation via an e-mail or an SMS. As Tom cannot provide neither of them to Alice, she will not be able to achieve her goal as a consequence of the context she acts in.

4

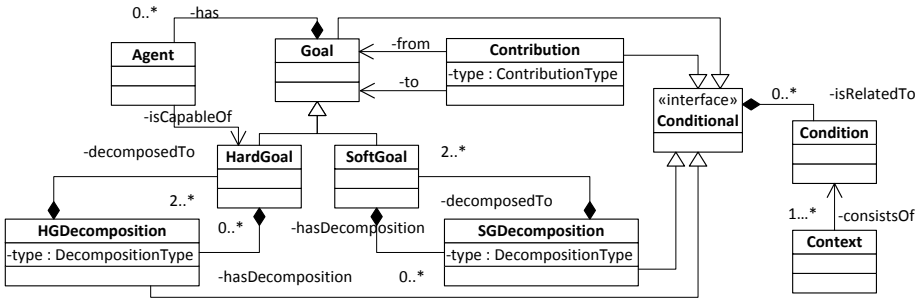


Fig. 3: The metamodel for the contextual variability of agents' goals.

3.2 From Conditional Elements to Commitments

In this section we are going to define a transformation process for the mapping of conditional elements to commitments. Initially, we define a dummy agent called *ContextAgent* which has no goals to achieve, and always adopts the role *Reasoner*. The purpose of this agent is to provide other agents with goals that belong to the set of its capabilities (set G_C), something that can be done by enriching the existing protocol with additional commitments. More than that, for each condition in the model we create a corresponding role that can be adopted by the agents that participate in the application.

The mapping from conditional contributions to commitments for the four possible types ($++S/D$ and $--S/D$) [1], along with the corresponding transformation steps are summarized in Table 1. Because of space limitations we will only discuss in detail the transformation process for the $--D$ conditional contribution. We start by adding two new goal nodes to the model, namely g'_s and g'_t, with the latter being a capability of the ContextAgent. Those two goal nodes are then connected to the goal model via two *unconditional* contributions, one of type $--D$ from g_s to g'_s, and a second one of type $++S$ from g'_t to g_t. Subsequently, the commitment $Commitment(Reasoner,r_{\text{cond}},g'_s,g'_t)$ is inserted in the protocol of the application. What remains to be proved is the soundness of the mapping from the $--D$ conditional contribution, to the rules that use commitments and unconditional contributions. In other words, we must show that when condition C_{cond} is true, and g_s is not satisfied, goal g_t is fulfilled.

Actually, if g_s is denied, g'_s is satisfied because of the existence of the $--D$ contribution from the former to the latter. This implies that if the agent adopts the role r_{cond} (this is the role that corresponds to C_{cond}), and because of the commitment previously specified, ContextAgent will provide it with g'_t which is one of ContextAgent's capabilities. Finally, the truth of g'_t has as a consequence the satisfaction of g_t goal node because of the $++S$ contribution introduced earlier. Hence, if g_s is denied and the agent adopts the role r_{cond}, g_t becomes true, while if r_{cond} is not adopted, g_t can not be satisfied as a consequence of the

[1] In the context of this paper we adopt the semantics of [1] for contributions

Table 1: Mapping conditional contributions to commitments.

Conditional Contribution	**Mapping to Commitments**
$g_s \xrightarrow[\{\text{cond}\}]{++S} g_t$	$Commitment(Reasoner, r_{\text{cond}}, g_s, g'_t)$ $g'_t \in G_C,\ g'_t \xrightarrow{++S} g_t$
$g_s \xrightarrow[\{\text{cond}\}]{--S} g_t$	$Commitment(Reasoner, r_{\text{cond}}, g_s, g'_t)$ $g'_t \in G_C,\ g'_t \xrightarrow{--S} g_t$
$g_s \xrightarrow[\{\text{cond}\}]{--D} g_t$	$Commitment(Reasoner, r_{\text{cond}}, g'_s, g'_t)\ ,\ g'_t \in G_C$ $g_s \xrightarrow{--D} g'_s\ ,\ g'_t \xrightarrow{++S} g_t$
$g_s \xrightarrow[\{\text{cond}\}]{++D} g_t$	$Commitment(Reasoner, r_{\text{cond}}, g'_s, g'_t)\ ,\ g'_t \in G_C$ $g_s \xrightarrow{--D} g'_s\ ,\ g'_t \xrightarrow{--S} g_t$

denial of g_s, as in this case the corresponding commitment cannot be applied. This means that by substituting the initial $--D$ *conditional* contribution with a commitment and an appropriate combination of unconditional contributions, we end up with a model that does not contain the initial conditional element but still has the same behavior as if it was part of the model.

Finally, the remaining two conditional elements, i.e. decompositions and goals, can be fully described by utilizing conditional contributions, and so, the same transformation process can also apply in those cases. The way a conditional decomposition, and an OR-child node are encoded as conditional contributions is illustrated in Fig. 4a and 4b respectively, while because of the duality between AND and OR-decomposition the former has been omitted.

4 Conclusion

In this paper we use goal models with conditional elements as a mean to capture the contextual variability of agents that participate in a multi-agent application. Consequently, we propose a set of rules for the transformation of those conditional elements to commitments and contributions. This transformation allows us to apply the same reasoning process introduced in [1] even in the presence of conditional goals, decompositions and contributions. Hence, we have succeeded in increasing the expressiveness of the metamodel introduced in [4], while at the same time the proposed reasoning techniques can still apply without changes.

5 Ongoing and future work

The extended metamodel presented in this paper is intended to be used for modeling policies that must apply in multi-layer systems at run-time. The formal representation of those policies, along with a proper reasoning engine, and in combination with advanced monitoring techniques, can then be utilized in order to check whether the system complies with the required policies or not.

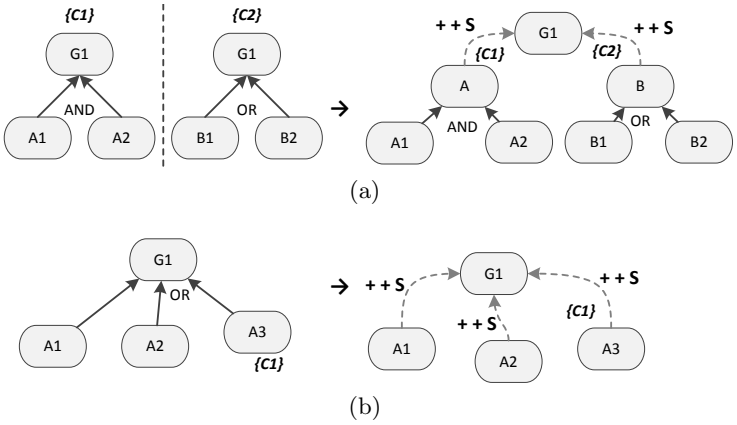


Fig. 4: Conditional a) decomposition ($G1$ is AND-decomposed if $C1$ holds, or OR-decomposed if $C2$ holds) and b) goal encoded as conditional contributions.

In our work we mainly focus on the former two parts, namely policy modeling, and reasoning, and we examine the applicability of probabilistic and fuzzy logic reasoners to the problem of policy validation.

An application of the proposed modeling is presented in [5]. The problem we are trying to solve in this paper is quite different, as we are interested in predicting the satisfaction of goals related to software project development. However, we show how a goal model with conditional modeling elements can be transformed to a first order logic knowledge base, which can then be used to perform probabilistic reasoning.

Acknowledgment This research is co-funded by the European Union (European Social Fund ESF) and Greek National funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: Heracleitus II. Investing in knowledge society through the European Social Fund.

References

1. Chopra, A.K., Dalpiaz, F., Giorgini, P., Mylopoulos, J.: Reasoning about agents and protocols via goals and commitments. In: AAMAS. (2010) 457–464
2. Ali, R., Dalpiaz, F., Giorgini, P.: A goal-based framework for contextual requirements modeling and analysis. Requir. Eng. **15**(4) (2010) 439–458
3. Lapouchnian, A., Mylopoulos, J.: Capturing contextual variability in *i** models. In: iStar. (2011) 96–101
4. Chopra, A.K., Dalpiaz, F., Giorgini, P., Mylopoulos, J.: Modeling and reasoning about service-oriented applications via goals and commitments. In: CAiSE. (2010)
5. Chatzikonstantinou, G., Kontogiannis, K.: A goal driven framework for software project data analytics. In: CAiSE. (2013) To appear.

Non-Functional Requirements Revisited

Feng-Lin Li[1], Jennifer Horkoff[1], John Mylopoulos[1], Lin Liu[2], Alexander Borgida[3]

1: Dept. of Information Engineering and Computer Science, University of Trento, Trento, Italy
2: School of Software, Tsinghua University, Beijing, China
3: Department of Computer Science, Rutgers University, New Brunswick, USA

Abstract. Goal-Oriented Requirements Engineering (GORE) is founded on the premise that functional and non-functional requirements (NFRs) are stakeholder goals to be fulfilled by the system-to-be. Moreover, functional requirements are "hard" goals with clear-cut criteria for fulfillment, while traditionally NFRs are usually "soft" goals (aka softgoals) lacking a clear-cut criterion for success. We argue against this distinction and in favor of a different one: traditional NFRs (e.g., security, reliability, performance, usability etc.) are requirements for ***qualities*** that existentially depend on the subject they qualify. We give examples in support of our argument, and sketch an abstract syntax and semantics for goal models that follow our proposal.

Keywords: Goal Model, Softgoal, Quality, Quality Constraint, Ontology

1 Introduction

The rise of goal orientation as a research paradigm for Requirements Engineering (RE) is founded on the premise that functional and non-functional requirements can be modeled and analyzed as (stakeholder) goals. According to this view, functional requirements are modeled as hard goals with a clear-cut criterion for fulfillment, while non-functional requirements (hereafter NFRs) are modeled as soft goals (aka softgoals) with no such criterion [1], hence their name. This paradigm served as research baseline for *i** [2], but has also enjoyed much broader attention within the RE community during the past 20 years.

In this position paper we challenge this orthodoxy. In particular, we argue that traditional NFRs, such as performance, reliability, maintainability, etc., are not softgoals, but rather requirements for *qualities* that existentially depend on the subject they qualify [3]. Moreover, as quality requirements, these kinds of NFRs are very special optative statements: they constrain the space of allowable quality values for their subject. This key insight, missing from earlier treatments of NFRs, influences in important ways both the abstract syntax and the semantics of requirements models.

To capture this missing existential dependence, we distinguish *quality requirements* from NFRs and promote them to first-class status in our framework. At the same time, to allow the "defuzzification" of vague quality requirements, we follow Techne [4], which made the satisfaction of softgoals measurable by operationalizing them as "quality constraints" – though at the time the term "quality" did not have the ontologically technical meaning to be introduced in this paper. Based on these, we revisit the requirements model and give examples in support of our proposal.

2 Quality Requirements

DOLCE [3] is an ontology that aims at capturing the ontological categories underlying natural language and human common sense. It provides a rich theory of qualities, which we adopt here. According to DOLCE, a ***quality*** is a particular which applies to a particular subject, and *inheres in* that subject, unable to exist independently from it. This existential dependence makes the semantics of qualities different from those of softgoals: we can only assess the satisfaction of a quality if its subject exists. For example, we cannot assess the security of the goal "*Message be Sent*" unless the message has been sent (this is also noted in [5]). However, this important existential dependence of qualities has not been captured in previous GORE techniques, such as *i** [2], the NFR framework [6], Tropos, and Techne [4].

The original proposal in DOLCE associates a particular quality, say *c*#1, of quality type *Cost*, to an individual, say *trip*#1, in class *Trip*; and then associates with *c*#1 the particular cost value, say 1000€ , in the *Cost* quality space *EuroValues*. We rephrase this by using a single, higher-order function *hasQualityValue*, which takes as argument the quality type *QT*, and returns a function that maps particular subjects to their quality values. Thus *hasQualityValue*(*QT*) can be used as in *hasQualityValue*(*Cost*)(*trip*#1) to obtain the cost value 1000€. Meanwhile, we overload this function to apply to a class/type *Subj*, by applying *hasQualityValue*(*QT*) to all instances of *Subj*, obtaining the set of quality values for all these individuals. E.g., *hasQualityValue*(*Cost*)(*Trip*) returns the set of all trips' costs. Note that in order to accommodate different degrees of accuracy, DOLCE views a quality space as consisting of regions with sub-regions. So a trip might have "*Low*" cost, with "*Very Low*" being a sub-region.

We view a ***quality requirement*** (*QR*) as a constraint over the region of expected quality values (e.g. low, fast, or high) of relevant *QT* (e.g. cost, speed, or performance) for a subject (type). On the basis of the above defined function, this can be formalized as $hasQualityValue(QT)(Subj) \subseteq RG$, where *QT* is a quality type, *Subj* is a class of subject individuals, *RG* is a possibly underspecified region of desired quality values. To make the quality region *RG* measurable, quality constraints (*QCs*) are introduced to precisely define its boundary [4][7]. For example, the quality requirement "*the Cost of Trip should be Low*", formally written as $hasQualityValue(Cost)(Trip) \subseteq Low\ Cost$, can be made more precise by specifying a *QC*: $Low\ Cost = \{y \mid 0 \leq y \leq 500€\}$.

On this view, we can express the vagueness of quality requirements using the desired regions of quality values (of corresponding qualities), which are fuzzy (e.g. low, fast). QRs can be made measureable by operationalizing them as quality constraints. Keep in mind that NFRs are not only quality requirements; e.g., the NFRs "*Response Time* ≤ *5*sec" and "*Take a Nice Trip*" can be modeled as hard goals and softgoals resp. The foundational concepts related to NFRs in our framework are defined as follows.

- A ***softgoal*** is a goal without a clear-cut definition of satisfaction [2], but an *independent* existence. Its satisfaction is determined by its refinements, which can include softgoals, hard goals and quality goals.
- A ***quality goal*** (*QG*) represents a quality requirement, which consists of a quality type, a desired quality region and a subject, with a different representation; e.g., a *QG* "*Low Cost*[*Trip*]" represents a *QR* "*the Cost of Trip should be Low*". A quality goal can be refined as quality goals or operationalized by a quality constraint. Its satisfaction is determined by its subject and refinements/operationalization.

- A ***quality constraint*** precisely defines a desired quality region in the quality space of a quality type and is achieved or denied by tasks. In our framework, quality constraints are metrics, but they are ontologically broader than metrics according to Techne [4], from which this concept is adopted.

3 A Framework for Goal Models with Quality Goals

To address the observed deficiency, we propose a revisited goal modeling framework on the basis of Techne [4], which extends *i** with quality constraints, preferences and inconsistency handling. We use extended EBNF (Extended Backus-Naur Form) to sketch an abstract syntax, wherein "+" means one or more, "$*$" means zero or more, "$\cdot$" means 'and', "|" means 'exclusive or', "$\rightarrow$" means substitution, and "$\rightarrow$" with a label at the top indicates particular relationship. We intend to describe a graphical notation and don't impose an order for elements on the right-hand side of the rules.

(1) $Goal \xrightarrow{is} Hardgoal \mid Softgoal \mid QualityGoal$
(2) $Hardgoal \xrightarrow{refine} Hardgoal^{+}$
(3) $Hardgoal \xrightarrow{operationalizeAs} Task^{+}$
(4) $Softgoal \xrightarrow{refine} Hardgoal^{*} \cdot Softgoal^{*} \cdot QualityGoal^{*}$
(5) $QualityGoal \xrightarrow{refine} QualityGoal^{+}$
(6) $QualityGoal \xrightarrow{operationalizeAs} QualityConstraint^{+}$
(7) $QualityGoal \rightarrow Quality'[Subj]$

Fig.1 The Abstract Syntax of the Revisited Goal Modeling Framework

The revisited requirements model is shown in Fig. 1 and visualized in Fig. 2. As the baseline, we use goals as a means to represent requirements (thus a quality requirement is now represented as a quality goal). *Quality Goal* is the key concept in our framework: it is expressed in the form of *Quality'*[*Subj*] (*Q'* [*Subj*] for short, e.g. "*Low Cost*[*Book Flight*]"), in which *Q'* is an ***intentional quality*** with both quality type and the desired region of quality values (e.g. *Low Cost*) and *Subj* is the type of subjects (e.g. *Book Flight*) that *Q'* inheres in; its operationalization, namely quality constraint, clearly specifies the region within the entire quality space.

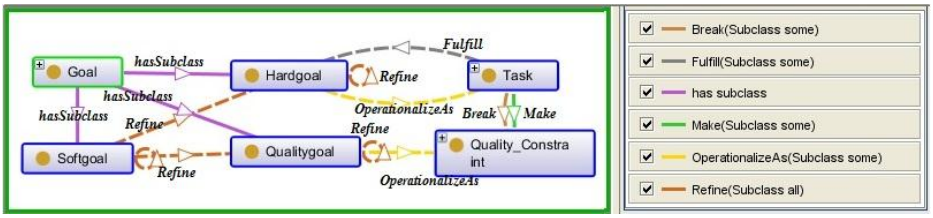


Fig.2 The Visual Representation of the Revisited Goal Modeling Framework

A travel example is used to illustrate the concepts in the framework. In this scenario, an employee *Lily* plans to take a leisure travel in Europe. As shown in Fig. 3, the top-level goal is a soft goal "*Take a Nice Trip*", which is refined to a hard goal "*Take a Trip*" and a quality goal "*Nice*[*Take a Trip*]". The hard goal is iteratively refined and finally operationalized as tasks; the quality goal is refined to other quality goals which are operationalized by quality constraints.

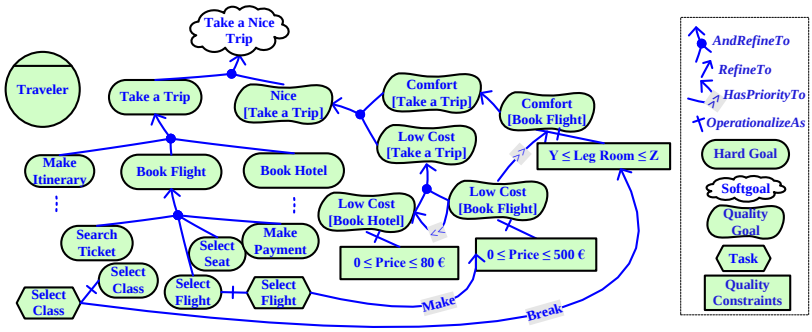


Fig.3 A Travelling Example

The refinement of a quality goal Q' [*Subj*] has two forms: via the intentional quality Q' and via the subject *Subj*. The refinement of Q' can be performed by following the existing quality type/dimension standards (e.g. ISO standards 9126 and more recently 25010), or using domain-specific knowledge. The refinement of *Subj* depends on either the hierarchy of hard goals or the ontology of subjects in Fig. 4. For example, refining by quality, *Nice* for subject *Take a Trip* usually includes *Low Cost*, *Comfort*, *Convenient* and *Timeliness*. In our example, only a subset of them is considered: "*Nice*[*Take a Trip*]" is refined as "*Low Cost*[*Take a Trip*]" and "*Comfort*[*Take a Trip*]". Refining by subject, taking a trip is composed of booking flight and hotel in the hard goal hierarchy, so the quality "*Low Cost*[*Take a Trip*]" is refined as "*Low Cost*[*Book Flight*]" and "*Low Cost*[*Book Hotel*]". In brief, the refinement of quality goals can be summarized as two rules:

$$(Rule \cdot A\colon Refine\ by\ Quality)Q' \xrightarrow{x} \{Q'_1, Q'_2 \dots Q'_n\} \Longrightarrow$$
$$Q'[Subj] \xrightarrow{x} \{Q'_1[Subj], Q'_2[Subj] \dots Q'_m[Subj]\}, (x = AND - refine | refine, 0 < m \leq n)$$
$$(Rule \cdot B\colon Refine\ by\ Subject)Subj \xrightarrow{x} \{Subj_1, Subj_2 \dots Subj_n\} \Longrightarrow$$
$$Q'[Subj] \xrightarrow{x} \{Q'[Subj_1], Q'[Subj_2] \dots Q'[Subj_m]\}, (x = AND - refine | refine, 0 < m \leq n)$$

In trying to understand the semantics of quality goals, we adopt a holistic perspective: a software system isn't just code, or code plus design and requirements; rather, it includes all the things that concern its existence, its makeup, development processes and history. Accordingly, quality goals can have any of these aspects as subjects. To clarify this, we give a simple ontology in Fig. 4, which defines the scope over which qualities apply. We intend for such subjects to cover both the problem and solution domain, allowing us to express both "as-is" and "to-be" quality requirements. Our ontology is not claimed to be complete and can be extended on demand.

(1) $Subject \xrightarrow{is} SoftwareSystem$
(2) $SoftwareSystem \xrightarrow{hasParts} Goal^+ \cdot Process^+ \cdot Thing^+$
(3) $Process \xrightarrow{hasParts} DevelopmentP \cdot BusinessP$
(4) $DevelopmentP \xrightarrow{hasParts} ElicitReqP \cdot DesignP \cdot ImplementationP \cdot TestP$
(5) $Thing \xrightarrow{hasParts} Artifact^+ \cdot Object^+$
(6) $Artifact \xrightarrow{hasParts} Architecture \cdot Behavioural \cdot Feature \cdot Code$

Fig.4 An Ontology for the Subjects of Qualities

In our proposal, the partial contribution links *help* and *hurt* are excluded, only *make* and *break* are preserved to represent the achievement of quality constraints by tasks.

We omit help and hurt because they force on us a four-valued logic (satisfied *S*, denied *D*, partially satisfied *PS* and partially denied *PD*) [8]. As such, potential conflicts arise: if a quality goal receives both help and hurt or two competitive quality goals are both partially satisfied/denied, it is problematic to identify which alternative solution (i.e. a set of tasks) better satisfies given hard goals, along with concerned quality goals [5].

Meanwhile, make and break allow us to deal with the achievements of quality constraints using a simpler two-valued logic (*S*/*D*). By incorporating preferences and priorities, our framework is able to facilitate the selection of the best alternative requirements. In our goal models, the "*HasPriotityTo*" links will be drawn from preferred quality goals to less preferred ones, and priorities will be assigned to concerned quality goals. In this way, we are able to obtain an ordered sequence of quality goals based on stakeholder preferences. Future work will focus on describing use of our framework for the requirements selection problem.

4 The Semantics of Quality Goals

Reasoning about the satisfaction of goals can be performed on goal models by using inference rules and label propagation algorithms [8][9]. The core of existing goal semantics can be summarized as [8]: if a goal *G* is AND-refined to a set of goals G_1, $G_2 \ldots G_n (n \geq 1)$, then *G* can be satisfied only if all the sub-goals are satisfied; if *G* is OR-refined, *G* will be satisfied if any of the sub-goals is satisfied; if a goal G_1 makes (resp. breaks) a goal G_2, then G_2 will be satisfied (resp. denied) if G_1 is satisfied.

In our framework, the semantics of quality goals are two-fold: (i) they existentially depend on subjects (ii) they constrain regions of quality values, which are clearly specified by quality constraints. The existential dependence indicates: if the subject *Subj* of a quality goal *QG* is not satisfied, then the satisfaction of *QG* is unknown. We use symbol *N* for this situation, which means "*no evidence*" (axiom 1). If the subject *Subj* is satisfied, then the satisfaction of *QG* is determined by its refinements (axiom 2 ~ 4). Take axiom (2) as an example, given that $QG = Q'[Subj]$, and *QG* is AND-refined to $G_1 \ldots G_i \ldots G_n$ $(1 \leq i \leq n)$, then *QG* will be satisfied if its subject *Subj* is satisfied and all of the sub-goals are achieved. The other axioms are structured similarly.

(1) $QG = Q'[Subj]: D(Subj) \Rightarrow N(QG)$

(2) $QG = Q'[Subj], QG \xrightarrow{AND-refine} \{QG_1 \ldots QG_i \ldots QG_n\}: S(Subj) \wedge (\wedge_i S(QG_i)) \Rightarrow S(QG) (1 \leq i \leq n)$

(3) $QG = Q'[Subj], QG \xrightarrow{AND-refine} \{QG_1 \ldots QG_i \ldots QG_n\}: S(Subj) \wedge (\vee_i D(QG_i)) \Rightarrow D(QG) (1 \leq i \leq n)$

(4) $QG = Q'[Subj], QG \xrightarrow{refine} \{QG_1 \ldots QG_i \ldots QG_n\}: S(Subj) \wedge (\vee_i S(QG_i)) \Rightarrow S(QG) (1 \leq i \leq n)$

(5) $QG = Q'[Subj], QG \xrightarrow{refine} \{QG_1 \ldots QG_i \ldots QG_n\}: S(Subj) \wedge (\vee_i D(QG_i)) \Rightarrow D(QG) (1 \leq i \leq n)$

In accordance with the ontology in Fig. 4, the subjects can be processes or things instead of goals. In such cases, we can speak of process execution or thing existence instead of goal satisfaction. Moreover, in our framework, we use AND-refinement and refinement, and avoid the use of explicit OR.

(6) $QG = Q'[Subj], QG \xrightarrow{OperationalizeAs} QC:$
$\forall x: Subj, S(x) \wedge hasQualityValue(QT, x) \subseteq QC \Rightarrow S(QG)$

(7) $QG = Q'[Subj],\ QG \xrightarrow{OperationalizeAs} QC, T \xrightarrow{make} QC: S(T) \Rightarrow S(QC);\ S(Subj) \wedge S(QC) \Rightarrow S(QG)$

(8) $QG = Q'[Subj],\ QG \xrightarrow{OperationalizeAs} QC, T \xrightarrow{break} QC: S(T) \Rightarrow D(QC); S(Subj) \wedge D(QC) \Rightarrow D(QG)$

The axioms (6 ~ 8) deal with the satisfaction of quality goals at the bottom of the quality goal hierarchy. A quality goal *QG* will be satisfied, if the actual quality value of each individual in *Subj* on quality type *QT* belongs to the region *RG* being specified by a quality constraint *QC* (axiom 6). Axiom (7 ~ 8) show the semantics of make and break: if a task *T* makes (resp. breaks) a quality constraint *QC*, then the corresponding quality goal *QG* of *QC* will also be satisfied (resp. denied). It is worth mentioning that to deny *QG*, we also need its subject *Subj* to be achieved; otherwise the satisfaction of *QG* is unknown.

5 Conclusions and Future work

In this paper, we identified from NFRs quality requirements and model them as quality goals. Accordingly, we proposed a revisited goal modeling framework, sketching an abstract syntax and semantics. There are several interesting and challenging problems open for discussion: How to properly and systematically handle ambiguous qualities (e.g. inexpensive and low cost)? How will tasks influence the quality values of the concerned qualities? How to incorporate context in our goal models? These will be the key concerns in our next steps.

References

1. J. Mylopoulos, L. Chung, and B. Nixon, "Representing and using nonfunctional requirements: A process-oriented approach," *Software Engineering, IEEE Transactions on*, vol. 18, no. 6, pp. 483–497, 1992.
2. E. S. Yu, "Towards modelling and reasoning support for early-phase requirements engineering," in *Requirements Engineering, 1997., Proceedings of the Third IEEE International Symposium on*, 1997, pp. 226–235.
3. C. Masolo, S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari, "WonderWeb Deliverable D18, Ontology Library (final)," 2009.
4. I. J. Jureta, A. Borgida, N. A. Ernst, and J. Mylopoulos, "Techne: Towards a new generation of requirements modeling languages with goals, preferences, and inconsistency handling," in *Requirements Engineering Conference (RE), 2010 18th IEEE International*, 2010, pp. 115–124.
5. J. Horkoff and E. Yu, "Comparison and evaluation of goal-oriented satisfaction analysis techniques," *Requirements Engineering*, pp. 1–24, 2012.
6. L. Chung, B. A. Nixon, and E. Yu, *Non-Functional Requirements in Software Engineering*, vol. 5. Kluwer Academic Pub, 2000.
7. I. J. Jureta, J. Mylopoulos, and S. Faulkner, "Revisiting the core ontology and problem in requirements engineering," in *International Requirements Engineering, 2008. RE'08. 16th IEEE*, 2008, pp. 71–80.
8. P. Giorgini, J. Mylopoulos, E. Nicchiarelli, and R. Sebastiani, "Reasoning with goal models," *Conceptual Modeling—ER 2002*, pp. 167–181, 2003.
9. J. Horkoff and E. Yu, "Analyzing goal models: different approaches and how to choose among them," in Proceedings of the 2011 ACM Symposium on Applied Computing, 2011, pp. 675–682.

Towards Know-how Mapping Using Goal Modeling

Daniel Gross[1], Arnon Sturm[1,2], Eric Yu[1]

[1]Faculty of Information, University of Toronto, Canada
[2]Department of Information Systems Engineering, Ben-Gurion University of the Negev, Israel
sturm@bgu.ac.il, {daniel.gross, eric.yu}@utoronto.ca

Abstract. In organizing the knowledge in a field of study, it is common to use classification techniques to organize concepts and approaches along dimensions of interest. In technology domains, an advance often appears in the form of a new way or method for achieving an objective. This paper proposes to use goal modeling to map the means-ends knowledge ("know-how") in a domain. A know-how map highlights the structure of recognized problems and known solutions in the domain, thus facilitating gap identification and prompting new research and innovation. We contrast the proposed goal-oriented approach with a claim-oriented approach, using Web Page Ranking as a sample domain.

Keywords: Knowledge mapping, Knowledge exploration, Goal-oriented

1 Introduction

The term *know-how* is generally used to refer to knowledge about how to accomplish something effectively and efficiently. While it is widely acknowledged that a great deal of know-how is tacit, the body of literature in each technical domain reflects the cumulative store of the articulated know-how for that domain. Mapping out the conceptual structure of a body of know-how facilitates learning about the domain. Practitioners can use a map to seek out solutions to problems, and compare strengths and weaknesses of alternate solutions. Researchers can use a map to uncover gaps and guide research directions.

Our research is motivated by the observation that at the core, know-how involves means-ends relationships. Most existing approaches for mapping knowledge, such as classification [1], citation graphs [1,3], concept maps [5] and claim-oriented argumentation [6, 7] do not give special attention to the means-ends relationship. Since the means-ends relationship is also at the core of goal modeling approaches such as i*, we are interested in exploring a goal-oriented approach to mapping know-how. In our preliminary investigation, we have applied goal-modeling to map know-how from published literature in several domains. In this paper we motivate our research with a brief comparison with ScholOnto, a claim-oriented argumentation framework used to describe and debate scholarly literature [6, 7].

2 Objectives

This research aims to eventually offer a framework that:

- Provides a language to construct a map of the high-level structure of problem-solution relationships in a domain, along with strengths and weaknesses of known solutions, thus facilitating identification of knowledge gaps,
- Requires little or no special training to contribute or dispute know-how, so as to accommodate a wide range of users from novice to expert,
- Supports multiple viewpoints with differing assumptions,
- Provides guidance on possible integration of know-how from diverse sources and viewpoints,
- Supports trust management based on know-how sources and contributors, for example by identifying influential sources,
- Provides tools for knowledge acquisition – for example, semi-automated extraction of means-end structures from knowledge sources (e.g., research papers or technical reports) to be weaved into the know-how maps,
- Supports reconciliation or integration of know-how from different domains, so as to facilitate interdisciplinary understanding and collaboration.

3 Scientific Contribution

To meet the needs of a wide range of users, we face a trade-off between ease of use and expressiveness. Greater language expressiveness contributes to more sophisticated reasoning capabilities, while ease of use is essential for attracting users.

We begin our exploration by adopting a subset of an existing goal-oriented language, i* [8], as a baseline language for mapping know-how. i* goals and softgoals are used to characterize problems in terms of desired domain outcomes and properties. Alternative solution approaches are modeled as i* tasks. Tradeoffs are revealed through contributions to softgoals. Contextual assumptions are modeled as beliefs (not illustrated in this paper).

In this paper we report on one exploration of the trade-off between ease of use and expressiveness, in which we compared a rich ontological argumentation approach, ScholOnto [6, 7] with our light-weight goal-oriented approach. A ScholOnto model consists of concepts and links. A concept could be a verbal description of a problem, data, a methodology, a theory, etc. Concepts connected by links represent claims. Although ScholOnto offers link types called “addresses” and “solves” (under the “Problem-Related” category) among its fairly large number of link types (Figure 1), its focus is on argumentation on claims rather than on the structure of know-how. Figure 2 shows a ScholOnto argumentation model taken from [6, 7]. The model captures some scholarly claims that appear in a paper on Page Ranking algorithms [2].

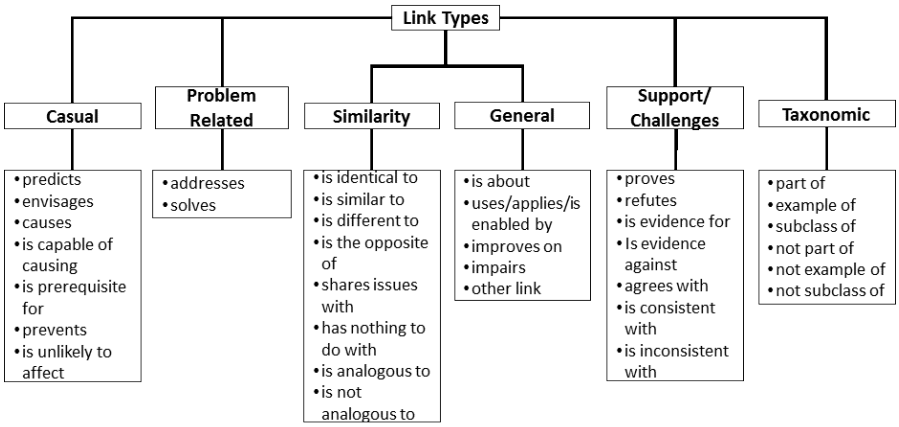


Figure 1: ScholOnto Link type Ontology [7]

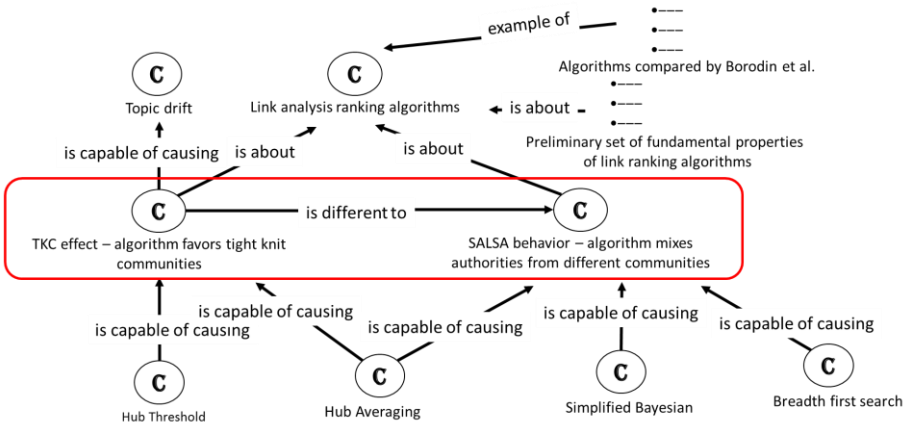


Figure 2: Argumentation Model Fragment using ScholOnto (adapted from [7])

In the study reported in [2], researchers submitted queries to different page ranking algorithms. They observed that pages that are part of tightly linked (page) communities show up higher when searched using some algorithms. They call this the Tight Knit Community (TKC) effect. Another family of algorithms favored pages drawn from different communities. This is referred to as SALSA behavior, as the search randomly goes back and forth between incoming and outgoing links of a selected pool of web pages. Authority is a measure of importance of a web page.

In Figure 2, the claim that there is a difference between these two types of behavior is expressed by the "is different to" link between the concepts "TKC effect – algorithm favors tight knit communities" and "SALSA behavior – algorithm that mixes authorities from different communities". Another claim states that a "Hub threshold" mechanism in a ranking algorithm "is capable of causing" the TKC effect. "Hub threshold" is a technique that discounts "hub" pages if they do not themselves have

certain authority ranking. The TKC effect in turn is capable of causing "topic drift", where a search term leads to unrelated results.

Figure 3: Contrasting page ranking algorithms using goals

Figure 3 shows a goal model representation of the Page Ranking analysis[1]. At the top, the goal graph indicates that the overall task of "Order page results from web query" should be addressed reliably, indicated by the quality goal "Reliable ordering." One way to achieve this is to "Rank pages according to authority" with the following associated quality goals: reducing "topic drift" (results off the query topic), increase results that are of relevant authority, and stability of results ranking with respect to small changes in the page links. To rank pages according to authority, two families of

[1] The model includes some of the concepts included in Figure 2, as well as a few other concepts drawn directly from [2].

algorithms are identified: Graph theoretic and Bayesian. These, in turn, can be designed to include specific features such as "Page base set", "Hub Threshold" and "Hub Averaging". Each of these alternatives affects some of the higher level design (soft-) goals.

Contrasting the goal model in figure 3 with the argumentation model in figure 2, we observe that argumentation model is primarily concerned with claims and their validity. The goal model, on the other hand, is focused on possible ways or methods for achieving goals, and how alternate methods contribute differently to various quality objectives. While the two models cover roughly the same domain, the know-how goal model is better suited to support design activities, such as seeking solution to problems, comparing alternatives, and making trade-offs.

We note that although the ScholOnto ontology includes "solves" and "addresses" as link types (Figure 1), these are merely used to document such relationships as claims (potentially refutable by another author). They are not meant to support analysis of goal achievement, unlike in goal-oriented modeling frameworks such as i*.

4 Conclusion

Today, there are a number of approaches that can be used to map the knowledge structure of a domain. However, they aim to cover knowledge structures in general with no special support for means-ends relationships or problem-solution reasoning.

Unlike ScholOnto, which offers a large number of link types, we considered a light-weight approach using a small number of concepts focusing on problem-solution structures. We aim to explore to what extent this light-weight approach can already be useful for the specialized purpose of know-how mapping.

This initial investigation suggests that a goal-oriented language, with limited expressiveness focusing on means-ends and problem-solution structures, may suffice for know-how mapping. Its compact representation could contribute to ease of use (to be validated), due to its minimal set of modeling constructs.

To-date we have explored a goal-oriented conceptualization of know-how in several domains and at different levels of abstraction, including Big Data, Goal-Oriented Software Architecting, Web Data Mining, and Agent-Oriented Software Engineering and found that the minimal set of concepts used by the goal-oriented approach allows us to identify problems and their associated qualities, solutions to these problems and their evaluation following the problem qualifier properties, and thus opportunities for innovation.

5 Future Work

We will continue to investigate the appropriateness and adequacy of representing domain know-how in terms of goal model structures. For added expressiveness, we are considering the incremental overlay of conceptual structures such as beliefs, assumptions and preconditions. We will explore modular structuring approaches, such as modules and intentional actor concepts.

D. Gross, A. Sturm, E. Yu

We will consider the use of actor boundaries as an abstraction mechanism to delineate knowledge communities (research communities and communities of practice). The aim is to provide a simplified and readily accessible view of the know-how of a domain to non-specialists, recognizing that the full details of in-depth solution considerations are of interest primarily to "insiders" of a specialist community. We are exploring how know-how mapping can help when attempting to bridge neighbouring domains, such as Requirements Engineering and Architectural Design.

We are developing guidelines and methodologies on how to approach the mapping of domains without prior knowledge of goal modeling. We need to address issues of readability and scalability of know-how maps expressed as goal-graphs. We are experimenting with tool support and visualization strategies. We plan to perform empirical studies with users to test the effectiveness of various know-how mapping approaches under different use cases, including getting acquainted with a domain using a know-how map, navigating the structure inside a know-how map, and extracting know-how from knowledge sources to add to an existing know-how map.

References

1. Aris, A., Shneiderman, B., Qazvinian, V., Radev, D.: Visual overviews for discovering key papers and influences across research fronts. J. Am. Soc. Inf. Sci., 60: 2219–2228(2009)
2. Borodin A., Roberts, G.O., Rosenthal, J.S., Tsaparas, P.: Finding authorities and hubs from link structures on the World Wide Web. In Proceedings of the 10th international conference on World Wide Web (WWW '01). ACM, New York, NY, USA, 415-42910th International World Wide Web Conference (WWW10), Hong Kong (2001)
3. Elkiss, A., Shen, S., Fader, A., Erkan, G., States, D., Radev, D.: Blind men and elephants: What do citation summaries tell us about a research article?. J. Am. Soc. Inf. Sci. 59: 51–62 (2008)
4. Kwasnik B.: The role of classification in knowledge representation and discovery. Library Trends 48, 22-47 (1999)
5. Novak J. D.,Cañas Al. J.: The Theory Underlying Concept Maps and How To Construct and Use Them, Institute for Human and Machine Cognition (2006).
6. Shum, S.B., Motta, E., Domingue, J.: ScholOnto: An ontology-based digital library server for research documents and discourse. International Journal of Digital Libraries, 3(3): 237-248 (2000)
7. Uren,V., Shum, S.B., Bachler, M., Li, G.: Sensemaking tools for understanding research literatures: Design, implementation and user evaluation, International Journal of Human-Computer Studies 64(5) 420-445 (2006)
8. Yu, E., Giorgini, P., Maiden, N., Mylopoulos J. (eds): Social Modeling for Requirements Engineering. Cambridge, MA: MIT Press. (2011)

Tool Fair Preface

Many tools have been created to facilitate modeling and analysis with *i** and related frameworks. The 2nd International iStar Tool fair aims to update community knowledge about the current offering of *i** tools. The Fair occurs as part of the 6th International *i** Workshop (iStar'13), collocated with CAiSE'13 in Valencia, Spain.

We received six tool submissions that were reviewed by three members of the iStar'13 programme committee, providing feedback and suggestions. All the proposals were considered interesting to the community, and were accepted in the form of a three-page description. The authors were also requested to either create or update the description of their tool available in the *i** wiki.

The Tool Fair is organized as a plenary session in the *i** workshop. It includes a short overview and summary of tool submissions made by the Tool Fair chairs, then short "lightening presentations" by each demo presenter, and finally the floor is opened for individual tool demos. This year the Tool Fair session includes presentation of a regular paper by Almeida et al. which performs a systematic comparison of *i** tools.

The iStar Tool fair includes an update of an existing modeling and analysis tool: the improved modeling and analysis of GRL models in the jUCMNav Tool is described by Amyot et al. The fair also includes descriptions of several tools which are new to the Tool fair, several of which support extensions of *i**. The CSRML Tool by Teruel et al. allows drawing of CSRML diagrams, an extension of *i** for Computer Supported Cooperative Work Systems. Paja et al.'s STS-Tool allows drawing and reasoning over STS-ml, an *i**-based security modeling language. Dalpaiz et al. provide the BIM tool, allowing for modeling and reasoning over the goal-based Business Intelligence Modeling language.

Several tools map *i** concepts to other languages or notations. The TAGOOn+ tool, provided by Najera et al., allows for organizational ontologies to be extracted from extended *i**/Tropos models. RE-Tools (Supakkul et al.) integrates and several popular RE modeling notations, including *i**, problem frames, KAOS, and UML.

We thank the authors for their valuable contributions, and look forward to seeing all of the tools in Valencia!

Jennifer Horkoff, *University of Trento, Italy*
iStar'13 Tool Fair Chair

CSRML Tool: a Visual Studio Extension for Modeling CSCW Requirements

Miguel A. Teruel, Elena Navarro, Víctor López-Jaquero, Francisco Montero, and Pascual González

LoUISE Research Group, University of Castilla-La Mancha (Spain)
{miguel, enavarro, victor, fmontero, pgonzalez}@dsi.uclm.es

Abstract. This work describes the CASE tool that provides support for CSRML (Collaborative Systems Requirements Modeling Language), an *i** extension for specifying CSCW systems requirements. The tool has been implemented as a Visual Studio 2012 extension by using the Visualization and Modeling SDK. It supports all the CSRML characteristics, such as the specification of collaborative tasks with Workspace Awareness features, as well as the management of actors, roles and groups of users involved in the system. Among other features, this tool supports also the automatic validation of the generated models, an integrated context-sensitive help system and automatic updates.

1 Introduction

A powerful CASE tool that supports modeling and validation of a Requirements Engineering (RE) language is a cornerstone for its success. That is the case of CSRML Tool (*a.k.a*. CSRMT), the tool that provides support for CSRML (Collaborative Systems Requirements Modeling Language) [2, 4], the *i**-based Goal-Oriented RE language developed to specify the special requirements of Computer Supported Cooperative Work (CSCW) systems, otherwise difficult or even impossible to model with classical RE techniques [3]. These special requirements are related to collaboration, communication and coordination (3C) tasks modeling, the actors, groups and roles management and, specially, the specification of Workspace Awareness (WA), which involves knowledge about, for example, *who* is available to collaborate, *what* are the other users doing now (or what they did in the past), *where* in the shared workspace are they working, *when* an artifact was modified or *how* a certain operation happens. Moreover, CSRML has been empirically validated by means of a family of experiments [5].

Because of all the CSCW features that CSRML is able to represent, it can be considered a graphically complex language, so that a powerful CASE tool is needed in order to guide the specification of a CSCW system by using this language. In order to lead that requirements specification, CSRML Tool was developed by implementing the CSRML metamodel with Microsoft Visualization and Modeling SDK, thus creating a Visual Studio 2012 (VS'12) extension that is presented in Section 2.

2 CSRML Tool

As previously mentioned, CSRML Tool 2012 is the CASE tool that provides support to CSRML (see Fig. 1). Currently, its development is on the stable version 2.0.130430, available at [1] along with a demo video and its documentation. This tool allows us to specify and validate a complete CSCW system by using the CSRML language, supporting the following functionalities:

- Full support for all CSRML features (e.g. WA support, 3C tasks or actors, roles and groups management)
- Specification of a complete CSCW system by means of the 5 different CSRML diagrams (GHD, SGD, RF, TRD and QFD [4]), guided by several VS'12 wizards
- Cross referencing of elements among the different system diagrams in order to preserve the model coherence (natively not supported by VMSDK and implemented by using the novel ModelBus technology)
- Diagrams validation in three different ways: design-time validation, meta-model validation and other potential sources of incoherence verification, such us recursive task and goal decompositions or duplicated references among models
- Full integration with VS'12, supporting automatic updates and communication with other Microsoft applications such as those included in Microsoft Office
- Version control of the generated models with Microsoft Team Foundation Server
- Context-sensitive help system for all the elements, relationships and diagrams, integrated with Microsoft Help Viewer 2.0. Tutorials for the most complex tasks are also included
- Expandable functionality by using Microsoft Managed Extensibility Framework

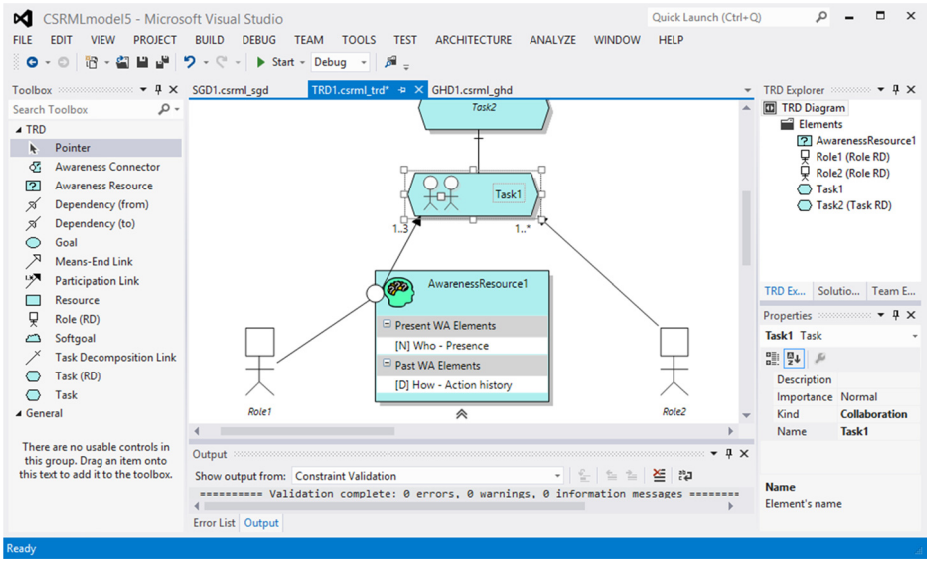


Fig. 1. CSRML Tool user interface

Additional details about the modeling elements, models editor, validation features as well as help and documentation can be found in [1]. Furthermore, it is worth noting that, because CSRML is based on *i**, this tool can also be used for specifying *i** requirements models, as it supports all its modeling elements and relationships, which are actually a subset of the CSRML ones.

3 Conclusions and Further Work

CSRML is a Goal-Oriented RE language, based on *i**, supporting the whole CSCW requirements modeling process. CSRML Tool is the software that supports the modeling of all the CSRML complex elements, relationships and diagrams. Therefore, CSRML and its supporting tool allow us to specify and validate CSCW systems, supporting 3C tasks modeling, WA specification and actors, roles and groups management.

In spite of being in a stable version, the tool is currently being modified to include the last CSRML features. In addition, usability testing has been recently performed in order to find any possible flaws regarding the tool user interface. Finally, this tool will be extended with new Model-Driven Development features in order to transform the RE specification to an analysis / design stage in a semi-automatic way.

Acknowledgements

This work has been supported by the grant insPIre (TIN2012-34003) and the FPU scholarship (AP2010-0259) from the Spanish Government.

References

1. Teruel, M.A.: CSRML Tool 2012, http://bit.ly/CSRMLTool (last accessed 02/05/2013).
2. Teruel, M.A., Navarro, E., López-Jaquero, V., Montero, F., González, P.: An extension of i * to Model Requirements for CSCW Systems Applied to Conference Preparation System with Collaborative Reviews. 5th International i* Workshop (iStar'11). pp. 84–89 , Trento (Italy) (2011).
3. Teruel, M.A., Navarro, E., López-Jaquero, V., Montero, F., González, P.: Comparing Goal-Oriented Approaches to Model Requirements for CSCW. Evaluation of Novel Approaches to Software Engineering. pp. 169–184 Springer-Verlag Berlin Heidelberg (2012).
4. Teruel, M.A., Navarro, E., López-Jaquero, V., Montero, F., González, P.: CSRML: A Goal-Oriented Approach to Model Requirements for Collaborative Systems. In: Jeusfeld, M., Delcambre, L., and Ling, T.-W. (eds.) 30th International Conference on Conceptual Modeling (ER'11). pp. 33–46 Springer Berlin Heidelberg, Berlin (2011).
5. Teruel, M.A., Navarro, E., López-Jaquero, V., Montero, F., Jaen, J., González, P.: Analyzing the Understandability of Requirements Engineering Languages for CSCW Systems: A Family of Experiments. Information and Software Technology. 54, 11, 1215–1228 (2012).

TAGOOn+: Generation and Integration of Organizational Ontologies

Karen Najera[1,2], Blanca Vazquez[1,2], Alicia Martinez[2], Anna Perini[3], Hugo Estrada[1,2], and Mirko Morandini[3]

[1] Fund of Information and Documentation for the Industry - INFOTEC, Mexico
{karen.najera, blanca.vazquez, hugo.estrada}@infotec.com.mx
[2] National Center of Research and Technological Development - CENIDET, Mexico
{amartinez}@cenidet.edu.mx
[3] Bruno Kessler Foundation - IRST, Center for Information Technology - FBK,Italy
{perini, morandini}@fbk.eu

Abstract. We present TAGOOn+, a tool that supports: 1) the automatic generation of organizational ontologies from models expressed with i*, Tropos and Service-Oriented i*, and 2) the automatic integration of those organizational ontologies with generic or domain ontologies.

Keywords: iStar, organizational modeling, ontology, ontology integration, Model-Driven Engineering.

1 Introduction

TAGOOn+[1] is a tool that automatically generates organizational ontologies and automates their integration with other ontologies. It has two main purposes:
1) It supports the automatic generation of organizational ontologies from organizational models expressed with i*, Tropos and Service-Oriented i* [2]. To do this, the ontological metamodel for these variants, called OntoiStar+ [3] has been developed. *Model-Driven Engineering (MDE)* ideas have been applied to transform organizational models into ontologies derived from OntoiStar+.
2) It supports the automatic integration of enriched organizational models with general or domain ontologies [4]. To do this, semantic annotation suggestions [4] are the guidelines to annotate organizational models with concepts coming from ontologies. The *iStarML format* [1] has been extended with the attribute "*sannotation*" [5] to store semantic annotations for each model element.

2 The TAGOOn+ tool

The overview of TAGOOn+ is presented in Fig. 1. TAGOOn+ receives as inputs: *(i)* an organizational model M_1 expressed with i*, Tropos or Service-Oriented i* represented in the iStarML format. M_1 can be a semantically annotated model, i.e., a model annotated with concepts from a generic or domain ontology O_D [4, 5]; and *(ii)* the ontology O_D, required if M_1 has been semantically annotated with O_D. The outputs of the tool are: *(i)* an organizational ontology O_{iStar},

[1] TAGOOn+ is developed by INFOTEC-CENIDET-FBK. The current version 1.0 is downloadable under GPL license from the tool homepage http://tagoon.semanticbuilder.com

2

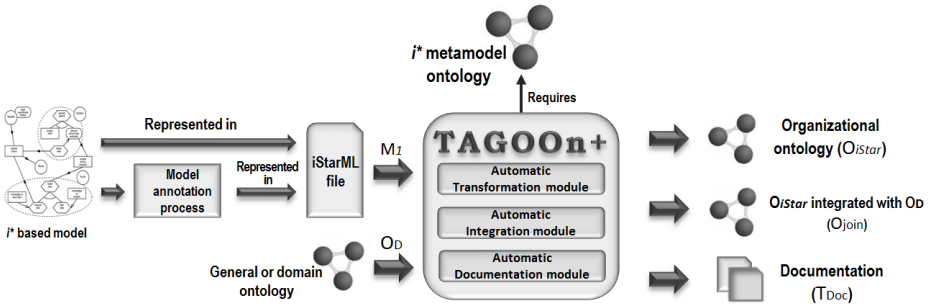


Fig. 1. Overview of TAGOOn+ tool

which represents the knowledge described in M_1; *(ii)* an ontology O_{join}, which integrates O_{iStar} and O_D and represents the knowledge described in M_1 and O_D; and *(iii)* a text file T_{Doc}, that describe the O_{iStar} and O_D integration process. These ontologies are described in the standard *Web Ontology Language OWL*.

TAGOOn+ has been developed using the environment of the Eclipse project and the Java programming language. It runs on Windows, Linux and Mac.

The tool is based on three main modules:

Automatic Transformation module. This module implements a transformation process following MDE ideas. Therefore, the ontological metamodel *OntoiStar+* has been developed in order to integrate (into an ontology) the i*, Tropos and Service-oriented i* construct definition [3]. Moreover, transformation rules have been defined in [2] to transform a model M_1 into an O_{iStar} ontology. O_{iStar} corresponds to the ontology OntoiStar+ instantiated with individuals that represent the knowledge depicted in M_1, including semantic annotations [5].

We ilustrate the transformation process with a short example. Let's assume that we have an i* based model describing a *disease detection process* and an ontology O_D that describes diseases and the parts of the human body. A task element of the model labeled as *Revise esophagus* is annotated with the concepts *swallow, stomach and animal-organ* taken from O_D. This task in iStarML format is represented as: `<ielement id="01" name="Revise esophagus" type="task" sannotation="swallow stomach animal-organ"/>`. After applying the transformation process, the task corresponds to an individual of O_{iStar} (see Fig. 2).

Automatic Integration module. This module integrates a semantic annotated model M_1 represented as ontology O_{istar} with the ontology O_D used to annotate M_1. It parses O_D to obtain its hierarchical structure and the description of each concept. Then, each individual of O_{istar} is related with one or more concepts of O_D through links of type *is_a*. In this way, O_{istar} and O_D are integrated in O_{join}. O_{join} contains the knowledge included in the semantically annotated i* based model M_1 integrated with the knowledge included in the ontology O_D. Following with the example (Fig. 2), the O_{istar} individual *Revise esophagus* is related by *is_a* links with the concepts *swallow, stomach and animal-organ*.

Automatic Documentation module. This module generates a text file that describes the O_{istar} and O_D integration process.

```
<j.1:Task rdf:about="http://www.cenidet.edu.mx/OntoiStar.owl#_01_Revise_esophagus">
<rdf:type rdf:resource="http://morpheus.cs.umbc.edu/aks1/ontosem.owl#swallow"/>
<rdf:type rdf:resource="http://morpheus.cs.umbc.edu/aks1/ontosem.owl#stomach"/>
<rdf:type rdf:resource="http://morpheus.cs.umbc.edu/aks1/ontosem.owl#animal-organ"/>
```

Fig. 2. A semantically annotated task element represented in OWL

A fragment of the generated documentation is: `type="task" name="Revise esophagus" annotation="swallow" description="To cause a bolus of food or drink to pass from the mouth/throat into the esophagus"`. The semantic annotations are useful to discover hidden information. In the example, annotations add information to the task about the fact that the esophagus is used to swallow, and it is related with the stomach and it is an organ.
Using the Semantic Web Rule Language SWRL, we could generate a simple rule expressing that a person is related with two elements stomach and esophagus, and these element are related too. If this person has problems with its stomach, then is necessary to revise the esophagus. The rule in SWRL would then be: `Person(?x)` $\wedge$ `stomach(?y)` $\wedge$ `esophagus(?z)` $\wedge$ `hasRelation(?y,?z)` $\wedge$ `hasProblems(?x,?y)` $\rightarrow$ `hasRevise(?z)`. This rule is a short example applying reasoning over the organizational knowledge.

3 Conclusion

We have presented the tool TAGOOn+. With this tool, we bring the advantages of ontologies such as querying and reasoning, to the organizational modeling domain. Moreover, as the organizational knowledge is represented in *OWL*, it could be available to be exploited and consumed in the Semantic Web by paradigms such as Linked Data. On the other hand, we provide the integration of organizational models enriched with semantic annotations with ontologies, which makes organizational knowledge clearer for humans and more accessible to machines.
Morever, we believe that a concept which integrates different model elements is a strong indicator to implement a new business services inside the organization, improving the understandability and expressiveness of an organizational model.

References

1. C. Cares, X. Franch, A. Perini, and A. Susi. Towards interoperability of i* models using istarml. *Computer Standards & Interfaces*, 33(1):69–79, 2011.
2. K. Najera. An ontology-based approach for integrating i* variants. Master's thesis, National Center of Research and Technological Development, Cuernavaca, Morelos, Mexico, 2011. `www.tagoon.semanticbuilder.com/NajeraThesis.pdf`.
3. K. Najera, A. Perini, A. Martínez, and H. Estrada. Supporting *i** model integration through an ontology-based approach. In *Fifth International i* Workshop (iStar'11)*, pages 43–48, 2011.
4. B. Vazquez. Enriching organizational models through semantic annotation. Master's thesis, National Center of Research and Technological Development, Cuernavaca, Morelos, Mexico, 2012. `www.tagoon.semanticbuilder.com/VazquezThesis.pdf`.
5. B. Vazquez, A. Martinez, A. Perini, H. Estrada, and M. Morandini. Enriching Organizational Model through Semantic Annotation. In *Proceedings of the Iberoamerican Conference on Electronics Engineering and Computer Science*, 2013.

Modeling and Tracing Stakeholders' Goals across Notations using RE-Tools

Sam Supakkul[1], Lawrence Chung[2], RJ Macasaet[3,4], Manuel Noguera[4], María Luisa Rodríguez[4], and José Luis Garrido[4]

[1] Sabre Inc., Southlake, Texas, USA
[2] The University of Texas at Dallas, Richardson, Texas, USA
[3] Pentathlon Systems Resources Inc., Manila, Philippines
[4] University of Granada, Granada, Spain
sam.supakkul@sabre.com, chung@utdallas.edu,
rjmacasaet@pentathlonsystems.com, {mnoguera,mlra,jgarrido}@ugr.es

Abstract. The ability to represent stakeholders' goals and their operationalizations, through tasks, resources, system requirements and specifications, helps better ensure that the projected system meets its intended goals. To offer such an ability, various notations have been developed, but somewhat independently of each other, each for its own concepts and with its own tool. As a result, it is difficult to establish end-to-end traceability among the various concepts. This paper presents RE-Tools - a toolkit towards an integrated modeling of such concepts, using *i**, the NFR framework, KAOS and Problem Frames. RE-Tools uses a meta-model approach to representing and integrating the notations, which includes shared, generalized meta-classes and cross-notational goal achievement contribution links. This approach is intended to allow for a uniform application of the Label Propagation Procedure in determining goal achievement across the supported notations, using either the open or closed world assumption.

1 General Information

Name	RE-Tools[1]
Version	2.0
Information	http://www.utdallas.edu/~supakkul/tools/RE-Tools
Frameworks	*i** [2], NFR Framework [3], KAOS [4], Problem Frames [5], UML
Purposes	Modeling and reasoning support, foundation for other tools [6]
Features	Integrated modeling environment, automated and integrated Label Propagation Procedure using open or closed world assumption, UML Profile-based meta-model, navigatable in-memory models, interchangeable XMI/XML file format, extension via plug-in and APIs, open-source, Windows OS supported.
Status	Publicly available since 2009, with around 1,300 downloads from around 65 countries to date, for use in teaching, research, and industrial practice, including Australia, Brazil, Canada, China, France, Italy, the Philippines, Spain, UK, and the US
Industry use	Internal use at Sabre and Pentathlon Systems Resources

2 Integrated Modeling and Tracing Goal Achievement

RE-Tools previously supported independent modeling of the supported notations and limited integration [1]. The tool has recently been enhanced to support a more comprehensive integration across multiple notations. Fig. 1 illustrates the key features of the enhancements, with the corresponding meta-classes shown in blue. Suppose the *Record ambulance location* Requirement in Problem Frames is determined to be satisficed (denoted by a check mark) by the system. Using the Label Propagation Procedure [7], the satisficing is propagated across the Make(++) contribution to derive that the *Ambulance location recorded* Goal in KAOS is also satisficed. The propagation is then repeatedly applied across the next contribution links in a cascading fashion upward the graph until the propagation cannot proceed any further because the needed information along the various paths is not available (e.g., the achievement of some sub-goals of an AND decomposition is undefined) or it has reached and evaluated the top-level Agents' goals (e.g., *Timeliness[Ambulance arrived at scene]* Softgoal).

To support the integrated modeling and reasoning, achievement contributions are represented as links between a parent Proposition and its offspring Proposition(s). Proposition is captured as a generalized meta-class in the meta-model, which is shared by the concepts in the supported notations (e.g., Hardgoal, Softgoal, BehavioralGoal, Task, Resource, Requirement). By applying the Label Propagation Procedure at the shared generalized Proposition meta-class level, RE-Tools performs reasoning about goal achievement generically and uniformly across the different notations.

3 Discussion and Future Work

The notion of (hard) goal achievement in i^* and KAOS are generally treated in an absolute sense (*satisfied* or (absolutely) *denied*), while the notion of Softgoal achievement in the NFR Framework is "good enough", reflecting the more subjective, interacting, relative nature (*satisficed*, *weakly satisficed*, *denied*, *weakly denied*). To support a more general and uniform label propagation, RE-Tools adopts and applies the weaker notion (satisficing) with an assumption that *satisficed* and (soft) *denied* could at least roughly be translated to the notions of *satisfied* and (absolutely) *denied* in some cases. However, additional research is required to better understand this approach, compare it with other integration approaches, as well as understand how to use multiple notations together.

Additionally, RE-Tools currently requires human intervention to trigger the automatic cascading label propagation process by first manually labelling the goal achievement of leaf-level nodes, which could be inaccurate or time-consuming. We are investigating ways to help alleviate these problems in some situations by integrating RE-Tools with other development notations and tools (e.g., BMPN [8] and URN [9]), and simulation tools [10] to automatically obtain the actual or simulated goal achievement of the leaf-level nodes. Furthermore, from the requirements modeling perspective, the tool may need to be extended to support other ontological concepts, such as quality constraints and domain assumptions.

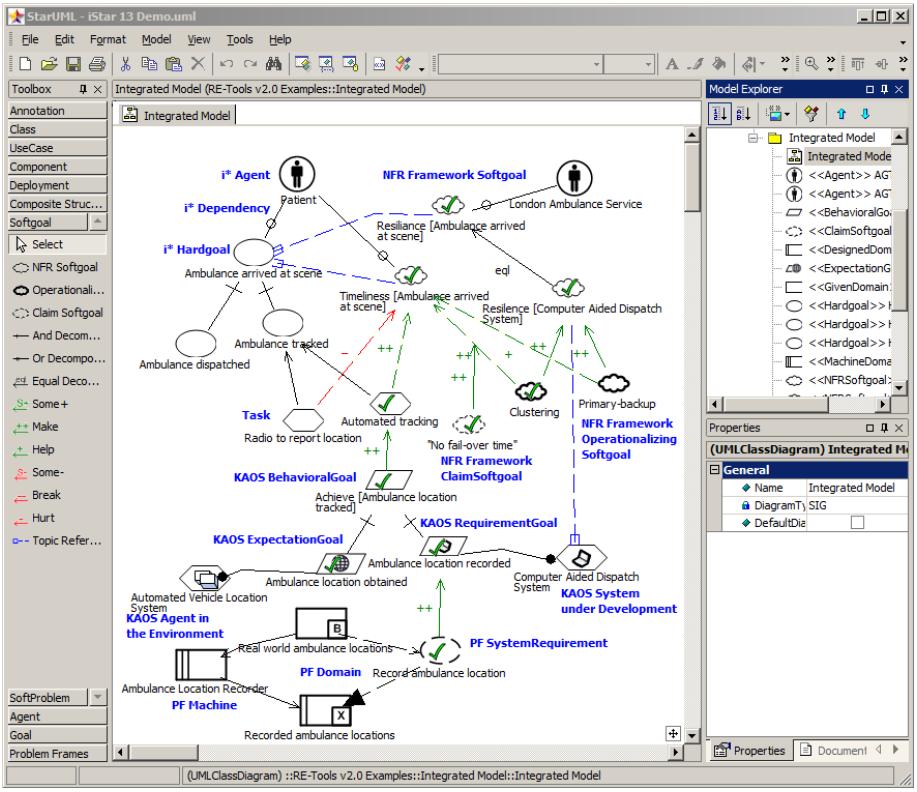


Fig. 1. A Sample Screenshot of an Integrated Model with Goal Achievement Reasoning

References

1. Supakkul, S., Chung, L.: The RE-Tools: A multi-notational requirements modeling toolkit. In: Requirements Engineering Conference (RE), 2012 20th IEEE Intl. (2012) 333 –334
2. Yu, E., Giorgini, P., Maiden, N., Mylopoulos, J.: Social Modeling for Requirements Engineering. The MIT Press (2011)
3. Chung, L., Leite, J.: On Non-Functional requirements in software engineering. In Borgida, A.T., Chaudhri, V.K., Giorgini, P., S.Yu, E., eds.: Conceptual Modeling: Foundations and Applications. (2009) 363–379
4. van Lamsweerde, A.: Requirements Engineering: From System Goals to UML Models to Software Specifications. Wiley (2009)
5. Jackson, M.: Problem Frames: Analyzing and Structuring Software Development Problems. Addison-Wesley (2000)
6. Supakkul, S., Hill, T., Chung, L., Tun, T., Leite, J.: An NFR pattern approach to dealing with NFRs. In: 18th IEEE Intl. Requirements Engineering Conf. (2010) 179–188
7. Chung, L., Nixon, B.A., Yu, E., Mylopoulos, J.: Non-Functional Requirements in Software Engineering. Kluwer Academic Publishers (2000)
8. Macasaet, R., Noguera, M., Rodrguez, M., Garrido, J., Supakkul, S., Chung, L.: A Requirements-Based Approach for Representing Micro-business Patterns. In: 7th IEEE Int'l Conference on Research Challenges in Information Science, IEEE (2013)
9. Amyot, D., Mussbacher, G.: URN: Towards a new standard for the visual description of requirements. In: Telecommunications and beyond: The BroaderApplicability of SDL and MSC. Springer (2003) 21–37
10. Hill, T., Supakkul, S., Chung, L.: Confirming and Reconfirming Architectural Decisions on Scalability: A Goal-Driven Simulation Approach. In: Proc. 8th Int'l. Workshop on System/Software Architectures, Springer (2009) 327–336

STS-Tool: Specifying and Reasoning over Socio-Technical Security Requirements

Elda Paja[1], Fabiano Dalpiaz[2], Mauro Poggianella[1], Pierluigi Roberti[1], and Paolo Giorgini[1]

[1] University of Trento, Italy – {elda.paja, mauro.poggianella, pierluigi.roberti, paolo.giorgini}@unitn.it
[2] University of Toronto, Canada – dalpiaz@cs.toronto.edu

Abstract. STS-Tool is the modelling and analysis support tool for STS-ml, our proposed actor- and goal-oriented security requirements modelling language for Socio-Technical Systems (STSs). STS-Tool allows designers to model an STS through high-level primitives, to express security constraints over the interactions between the actors in the STS, as well as to derive security requirements once the modelling is completed. The tool features a set of automated reasoning techniques for (i) checking if a given STS-ml model is well-formed, and (ii) determining if the specification of security requirements is consistent, that is, there are no conflicts among security requirements. We have implemented these techniques using disjuntive datalog programs.

1 The Socio-Technical Security modelling language

The *Socio-Technical Security modelling language* (STS-ml) [1] is an i^* based security requirements modelling language. STS-ml includes high-level organisational primivites such as actor, goal, delegation, etc. A distinguishing feature of STS-ml is the ability to relate security requirements to *interactions*: actors' security needs constrain the interactions they enter into with other actors. Security requirements are mapped to *social commitments* [3]—contracts among actors—that actors in the STS shall comply with at runtime.

STS-ml modelling uses three complementary views, in which the analyst examines different types of interactions among actors.

The formal semantics of STS-ml [2] defines the behavior of STS-ml concepts and relationships, allowing to perform: (i) well-formedness analysis to determine if the model complies with well-formedness rules that are set to preserve the semantics of the STS-ml primitives (e.g., decompositions are not cyclic), and (ii) security analysis, i.e., if there are potential conflicts of security requirements.

2 STS-Tool

STS-Tool is the modelling and analysis support tool for STS-ml. It is an Eclipse Rich Client Platform application written in Java, it is distributed as a compressed archive for multiple platforms (Win 32/64, Mac OS X, Linux), and it is

freely available for download from `http://www.sts-tool.eu`. The website includes extensive documentation including manuals, video tutorials, and lectures. STS-Tool has the following features:

- *Diagrammatic*: the tool enables the creation (drawing) of diagrams. Apart from typical create/modify/save/load operations, the tool also supports:
  - Providing *different views* on a diagram, specifically: *social view*, *information view*, *authorisation view*. Each view shows specific elements and hides others, while keeping always visible elements that serve as connection points between the views (e.g., roles and agents). Inter-view consistency is ensured by for instance propagating insertion or deletion of certain elements to all views.
  - Ensuring diagram validity (online): the models are checked for syntactic/ well-formedness validity while being drawn.
  - Exporting diagrams to different file formats (png, jpg, pdf, svg, etc.).
- *Automatic derivation of security requirements*: security requirements are generated from a model as relationships between a *requester* and a *responsible* actor for the satisfaction of a *security need*. Security requirements can be sorted or filtered according to their different attributes.
- *Automated reasoning*
  - *Offline well-formedness analysis*: some well-formedness rules of STS-ml are computationally too expensive for online verification, or their continuous analysis would limit the flexibility of the modelling activities. Thus, some analyses about well-formedness are performed upon explicit user request. In Fig. 1, offline well-formedness analysis has found no errors.
  - *Security analysis*: verify (i) if the security requirements specification is consistent—no requirements are potentially conflicting; (ii) if the diagram allows the satisfaction of the specified security requirements. This analysis is implemented in disjunctive Datalog and consists of comparing the possible actor behaviors that the model describes against the security requirements. The results are enumerated in a tabular form below the diagram, and rendered visible on the diagram itself when selected (see Fig. 1). A textual description provides details on the identified conflicts.
- *Generating requirements documents*: the modelling process terminates with the generation of a *security requirements document*, which supports the communication between the analyst and stakeholders. This document is customisable: the analyst can choose among a number of model features to include in the report (e.g., including only a subset of the actors, concepts or relations he or she wants more information about). The diagrams are explained in detail providing textual and tabular descriptions of the models. An example report is provided in [3].

The current version of STS-Tool (v1.3.1) is ready for public use. This version of the tool is the result of an iterative development process, having been tested on multiple case studies and evaluated with practitioners [4] in the scope of the

[3] `http://www.sts-tool.eu/Documentation.php`

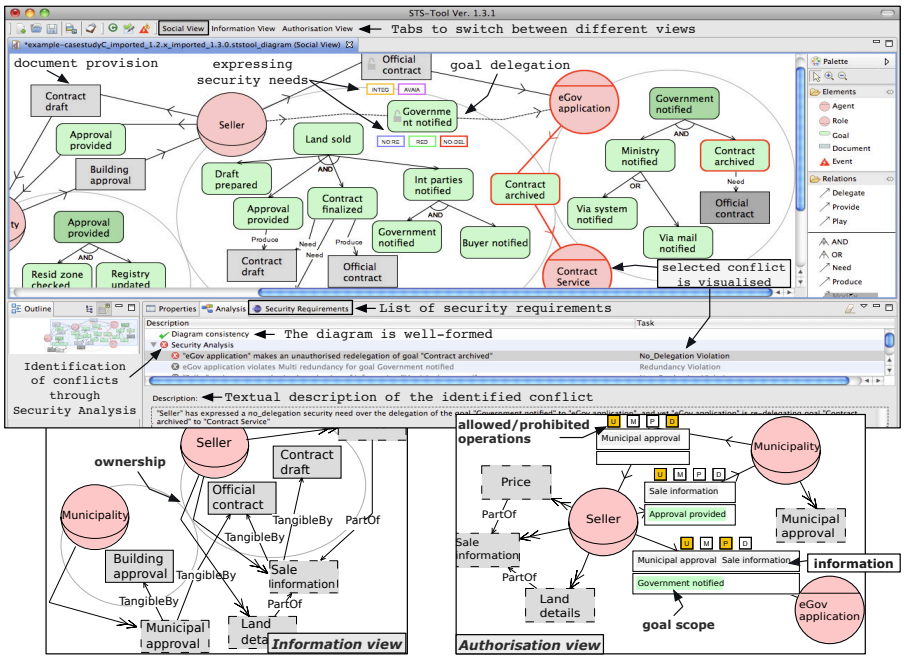


Fig. 1: STS-Tool screenshot: multi-view modelling, automatic derivation of security requirements, and visualisation of security analysis results

FP7 European Project Aniketos [4]. It has proven suitable to model and reason over models of a large size from different domains [2], such as eGovernment, Telecommunications, and Air Traffic Management Control.

Acknowledgments. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant no 257930 (Aniketos) and 256980 (NESSoS).

References

1. F. Dalpiaz, E. Paja, and P. Giorgini. Security requirements engineering via commitments. In *Proc. of STAST'11*, pages 1–8, 2011.
2. E. Paja, F. Dalpiaz, and P. Giorgini. Identifying conflicts in security requirements with STS-ml. TR DISI-12-041, University of Trento, `http://disi.unitn.it/~paja/tr-identifying-sec-conflicts.pdf`, 2012.
3. M. P. Singh. An ontology for commitments in multiagent systems: Toward a unification of normative concepts. *Artificial Intelligence and Law*, 7(1):97–113, 1999.
4. S. Trösterer, E. Beck, F. Dalpiaz, E. Paja, P. Giorgini, and M. Tscheligi. Formative user-centered evaluation of security modeling: Results from a case study. *IJSSE*, 3(1):1–19, 2012.

[4] `http://www.aniketos.eu`

BIM-Tool: Modeling and Reasoning Support for Strategic Business Models

Fabiano Dalpiaz[1], Daniele Barone[1], Jennifer Horkoff[2], Lei Jiang[1], and John Mylopoulos[1]

[1] University of Toronto, Canada – dalpiaz,barone,leijiang,jm@cs.toronto.edu
[2] University of Trento, Italy – horkoff@disi.unitn.it

Abstract. The BIM-Tool provides graphical modeling and analysis support for the Business Intelligence Model (BIM). BIM-Tool is a standalone application built on top of Eclipse. The tool supports two kinds of automated reasoning: (1) bottom-up "what-if" analysis: given input labels about some elements of a BIM model (for example, success/failure for leaf goals), do these propagate to other elements in the model?; and (2) top-down "is it possible?" analysis: is there a plan that leads to the satisfaction of goals, occurence/non-occurence of situations? BIM-Tool answers these questions through an encoding of BIM models in disjuntive datalog programs.

1 The Business Intelligence Model

The *Business Intelligence Model* (BIM) [1,2] is a goal-oriented language for modeling organizational strategies. BIM relies on a set of modeling primitives that decision makers are familiar with, such as goal, task/process, indicator, situation, and influence relations among them. BIM is intended to support the notions from SWOT (*Strengths*, *Weaknesses*, *Opportunities*, *Threats*) analysis by modeling the internal and externals factors (situations) that are (un)favorable for fulfilling strategic business goals.

Some of the primitives of BIM (goals, tasks, refinement) are adopted from *i**. Unlike *i**, BIM does not support strategic dependencies, for BIM focuses on the high-level strategic goals of the organization, and does not ascribe goals to specific actors. BIM does not distinguish between hard- and soft-goals; rather, it includes measurable indicators that determine the degree of satisfaction of goals.

BIM is endowed with a formal semantics [3] that enables a variety of automatic reasoning techniques [2]: (i) goal analysis (both top-down and bottom-up), (ii) probabilistic evaluation of strategies, and (iii) reasoning with composite and qualitative indicators.

2 BIM-Tool

The BIM-Tool aims to provide a comprehensive graphical modeling and analysis support for BIM. BIM-Tool is a standalone Rich Client Platform application built on top of Eclipse that exploits the Eclipse Graphical Modeling Project (GMP) framework [1]

[1] http://www.eclipse.org/modeling/gmp/?project=gmf-tooling

for metamodel-based development of graphical modeling environments. The analysis procedures are implemented through an encoding in disjunctive datalog programs.

The main features of BIM-Tool are as follows:

- Creating graphical models using the latest version of BIM [2]. The models are checked for syntactic validity while the elements and links are drawn, through the evaluation of OCL constraints.
- Automated reasoning about "what-if" scenarios using the bottom-up algorithms described in [3]. Given evidence (labels) about the satisfaction, denial, and pursuit of BIM elements (goals, situation, indicators, domain assumptions), our analysis propagates the input evidence through refinement and influence relationships in the model. The output is shown in a window at the bottom of the application.

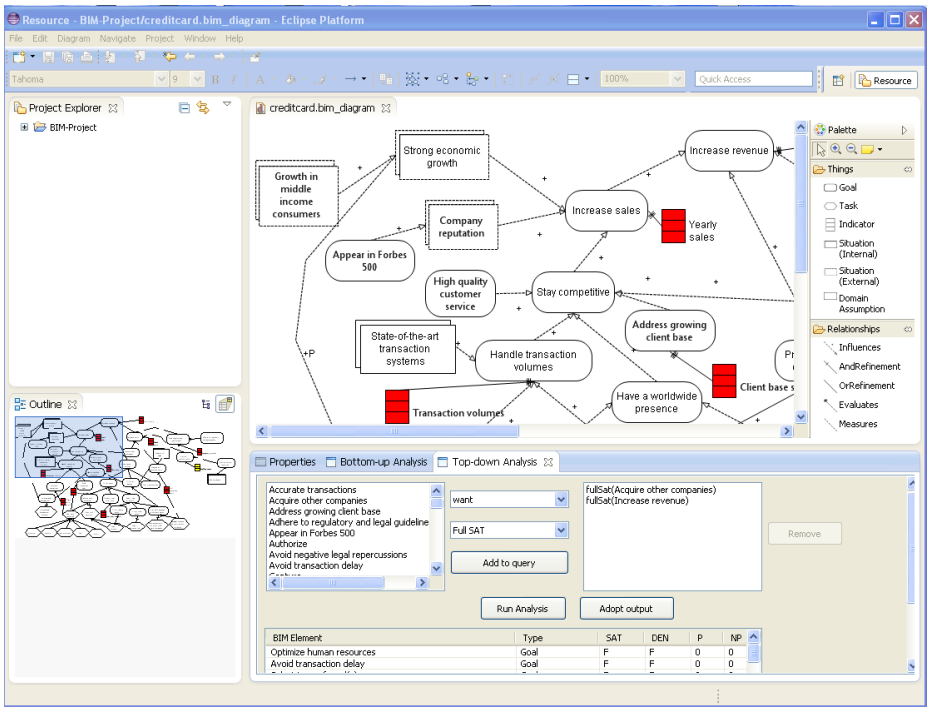


Fig. 1. BIM-Tool screenshot: running a top-down "is it possible?" analysis

- Automated reasoning about "is it possible?" scenarios. The analyst specifies a query that is expressed as a logical conjunction of BIM elements that the analyst wants to partially/totally satisfy/deny. The analyst can also specify that she wants to avoid the partial/total satisfaction/denial of some of the elements in the query. The tool returns a possible strategy—evidence and pursuit assignments—that satisfies the query, if such a strategy exists. Figure 1 shows the tool in action, while answering

a query of the "is it possible?" type. The results are shown in the table at the very bottom of the screenshot.
– Supporting the organization of models in projects, zooming, navigating the diagrams, concurrent handling of multiple diagrams, and exporting diagrams to a variety of image formats (both raster and vector). These functionalities are provided by including appropriate Eclipse plugins.

The tool is publicly available from the website `www.cs.toronto.edu/~jm/bim/`. Currently, the released versions support Microsoft Windows (both 32 and 64 bits) and Linux distributions (32 bits). The website includes basic usage tutorials through screencasts. One screencast is about how to model BIM diagrams, while other two screencasts show the two supported reasoning techniques. Readers can refer to existing BIM publications for information about the BIM language and reasoning [3,2,1].

3 Limitations and Outlook

The tool is intended to be a proof-of-concept prototype. As such, it can be freely downloaded and used, but it is not ready for industrial adoption. Additionally, BIM-Tool is currently supporting only a subset of the reasoning techniques developed for and adapted to the BIM language.

Our future work includes addressing current limitations of the tool, making it more robust for usage by practitioners (extensive testing is required), exploring interoperability with other tools (e.g., importing goal models from other tools), providing additional tutorials and more extensive documentation about the tool, and supporting the Mac OS operating system.

Acknowledgements

This work has been partially supported by by the Natural Sciences and Engineering Research Council (NSERC) of Canada through the Business Intelligence Network.

References

1. Daniele Barone, Thodoros Topaloglou, and John Mylopoulos. Business intelligence modeling in action: a hospital case study. In *Advanced Information Systems Engineering*, pages 502–517. Springer, 2012.
2. Jennifer Horkoff, Daniele Barone, Lei Jiang, Eric Yu, Daniel Amyot, Alex Borgida, and John Mylopoulos. Strategic business modeling: representation and reasoning. *Software & Systems Modeling*, pages 1–27, 2012.
3. Jennifer Horkoff, Alex Borgida, John Mylopoulos, Daniele Barone, Lei Jiang, Eric Yu, and Daniel Amyot. Making data meaningful: The business intelligence model and its formal semantics in description logics. In *On the Move to Meaningful Internet Systems: OTM 2012*, pages 700–717. Springer, 2012.

Improved GRL Modeling and Analysis with jUCMNav 5

Daniel Amyot, Rouzbahan Rashidi-Tabrizi, Gunter Mussbacher
School of Electrical Eng. and Computer Science, University of Ottawa, Canada
damyot@eecs.uottawa.ca, rouzbahan.r.t@gmail.com, gunterm@eecs.uottawa.ca

Jason Kealey and Etienne Tremblay
JUCM Software Inc., Canada
jkealey@jucm.ca, etremblay@jucm.ca

Jennifer Horkoff
DISI, University of Trento, Italy
horkoff@disi.unitn.it

Abstract. jUCMNav is an open-source Eclipse tool for modeling and analyzing stakeholder goals, scenarios, and requirements with the User Requirements Notation (URN) standard. This paper gives a brief overview of this tool, with an emphasis on recent improvements targeting URN's Goal-oriented Requirement Language (GRL) found in version 5.x.

Keywords: GRL, jUCMNav, URN.

1 Overview of the User Requirements Notation and jUCMNav

The User Requirements Notation (URN) integrates (i) the Goal-oriented Requirement Language (GRL) for stakeholder objectives and decision rationales and (ii) Use Case Map (UCM) for scenarios and business processes combined with architectural components. One important feature of GRL is the support for strategies, which define initial satisfaction values (qualitative or quantitative) for some intentional elements in a goal model that are propagated to other elements of the model (including actors) through various evaluation algorithms [5]. GRL and UCM have been used individually and together for more than a decade, in dozens of application areas [2]. The second version of the URN standard was released in October 2012 [6], four years after the first release, with several major improvements to GRL that include:

- *Indicators* to handle real-life values (beyond simple satisfaction values) in goal models. Indicators (symbol: ⟺) convert real-life values into GRL satisfaction values based on linear extrapolation or based on a mapping table.
- *Strategy inclusion* to improve the reuse, consistency, and maintainability of large collections of GRL strategies.
- *Contribution changes* to enable the description of sets of modifications to contribution weights in a GRL model (e.g., from different modelers).
- *Actor importance values* to better enable tradeoffs between strategies.

jUCMNav is an Eclipse-based environment for modeling and analysis with URN. Its development started in 2005, and an overview of version 4.x was given at iStar'2011 [3]. This is a mature tool that has been used in academia (for teaching and research), in industry, and even in government agencies for many years, by thousands of people [2], on academic systems and on real ones. The tool can be installed from its Wiki site [7], where tutorials, examples, and documentation are also available.

Primary features include the editing of URN models with palettes and context-sensitive menus. The tool prevents the creation of syntactically incorrect models, but it also offers users the chance to create, select, and check their own correctness/consistency rules, written in OCL. GRL/UCM elements can also be linked, stereotyped, and grouped, hence enabling the tailoring of the notation to particular domains (e.g., for *i**-like modeling [1], or for legal compliance [9]). Copy/paste of model portions, multiple undo/redo, and many navigation facilities contribute to the usability and scalability of the tool (e.g., models composed of hundreds of diagrams and thousands of elements). On the analysis side, UCM scenarios can be defined, run, and transformed to sequence diagrams. GRL strategies can be defined, evaluated (see Figure 1), and imported/exported to CSV files. Reports (in HTML, PDF, and RTF) can be generated to summarize models and analysis results. Models can also be exported to DOORS. Advanced features for indicators, performance, and aspect modeling can each be disabled to simplify the user interface for beginners.

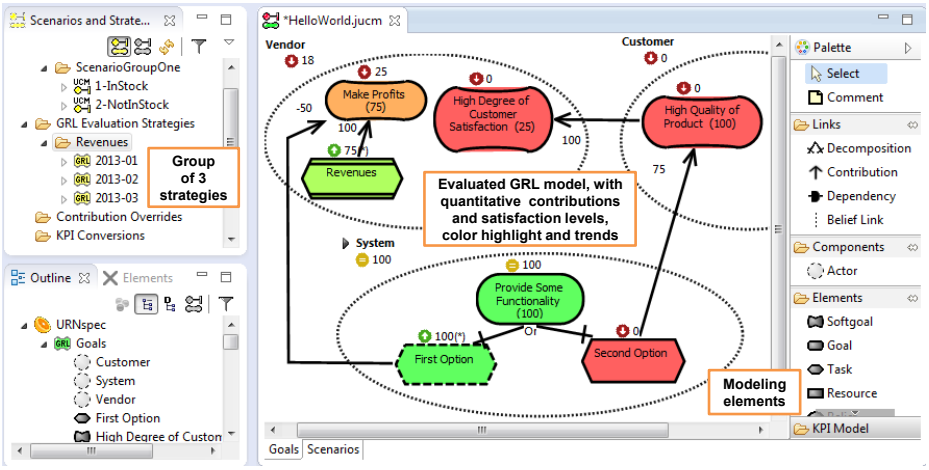


Figure 1, jUCMNav 5, with GRL strategy and GRL trends for group "Revenues".

2 Recent Developments in Version 5 of jUCMNav

The fifth major release of jUCMNav contains many important improvements, especially for GRL. The tool now supports the new concepts introduced in the second release of the standard. While quantitative indicators were already available for modeling and analysis [3], jUCMNav now also supports *qualitative indicators* (illustrated in [9]), which enable modelers to provide a mapping between enumerated domain-specific values (e.g., bad, average, good, excellent) and quantitative GRL values (0,

33, 67, 100) or qualitative GRL labels (e.g., WeaklySatisfied and Satisfied). *Strategy inclusion* and *contribution changes* are well illustrated in [4], together with advanced analysis features such as *strategy differences* (computing and visualizing the differences between the evaluations of two GRL strategies), *model differences* (highlighting the differences between two versions of a model), and *sensitivity analysis* (based on ranges of contribution or satisfaction values in strategies, instead of on specific values). Note that recent experience in using GRL for modeling laws [4,9] also led us to introduce a new *alternative GRL satisfaction scale* ([0..100], in addition to the standard [-100..100] scale), considered as more intuitive by many users. Also, given that laws in Canada are written in French and English, jUCMNav now supports *bilingual models*, where the labels and descriptions of elements can be switched from one language to the other (in addition to having the tool's interface in both languages).

Recent additions also include: i) simpler and more efficient UI to create, delete, and navigate URN links, ii) new capabilities to show the actors or intentional element related (through links or inclusion) to another actor/element (to various depths), iii) the possibility to define possible stereotypes and the type of model elements they can apply to, with appropriate pop-up menus to apply these stereotypes (useful in the context of [1,9]), iv) OCL rules and improved propagation algorithms that support *goal model families*, which enable the modeling and analysis of many variants in one GRL model [8], and v) the computation and visualization of *trends* (up, stable, down, varying, and insufficient data, see Figure 1) based on a sequence of strategies in a strategy group. Trends are also included for GRL elements in reports.

jUCMNav is still evolving. Future plans include further usability improvements, a textual syntax for URN, and better support for aspect-oriented extensions to GRL.

References

1. Amyot, D., Horkoff, J., Gross, D., Mussbacher, G.: "A Lightweight GRL Profile for i* Modeling". In: *RIGiM 2009*, ER Workshops. LNCS 5833, Springer, pp. 254–264 (2009)
2. Amyot, D. and Mussbacher, G.: "User Requirements Notation: The First Ten Years, The Next Ten Years". *Journal of Software (JSW)*, Vol. 6, No. 5, pp. 747–768 (May 2011)
3. Amyot, D., Mussbacher, G., Ghanavati, S., and Kealey, J.: "GRL Modeling and Analysis with jUCMNav". In: *iStar 2011*, CEUR-WS, Vol-766, pp. 160–162 (August 2011)
4. Amyot, D., Shamsaei, A., Kealey, J., Tremblay, E., Miga, A., Mussbacher, G., Alhaj, M., Tawhid, R., Braun, E., and Cartwright, N. "Towards Advanced Goal Model Analysis with jUCMNav". In: *RIGiM'12*, ER Workshops. LNCS 7518, Springer, pp. 201–210 (2012)
5. Amyot, D., Ghanavati, S., Horkoff, J., Mussbacher, G., Peyton, L., and Yu, E.: "Evaluating Goal Models within the Goal-oriented Requirement Language". *International Journal of Intelligent Systems*, John Wiley & Sons, Vol. 25, Issue 8, pp. 841–877 (2010)
6. International Telecommunication Union: *Recommendation Z.151 (10/12), User Requirements Notation (URN) – Language definition*, Geneva, Switzerland (2012) http://www.itu.int/rec/T-REC-Z.151/en
7. jUCMNav, version 5.x, University of Ottawa, http://softwareengineering.ca/jucmnav
8. Shamsaei, A., Amyot, D., Pourshahid, A., Yu, E., Mussbacher, G., Tawhid, R., Braun, E., and Cartwright, N.: "An Approach to Specify and Analyze Goal Model Families". In: *System analysis and Modeling: Theory and Practice*, LNCS 7744, Springer, pp. 34–52 (2012)
9. Tawhid, R. et al.: "Towards Outcome-Based Regulatory Compliance in Aviation Security". In: *RE 2012*. IEEE CS, pp. 267–272 (2012)

a CEUR Workshop Proceedings, ISSN 1613-0073

http://ceur-ws.org/Vol-978