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.