Model-Driven Visual Requirements Engineering

Helge Solheim[1], Frank Lillehagen[1], Sobah A. Petersen[1], Håvard Jørgensen[1], Maria Anastasiou[2]
[1]Troux Technologies, [2]Intracom
[1]{hsolheim, flillehagen, hjorgensen}@troux.com, [2]mana@intracom.gr

Abstract

This paper describes the requirements handling process of a set of research projects, the ATHENA IP, and how this process is supported by modeling and visualizing requirement structures. First, users and external stakeholders register requirements through easy-to-use web interfaces. Then developers and managers utilize an integrated modeling tool for visual classification, analysis, elicitation, and selection of requirements. In this paper, we show how advanced visualization techniques improve and support state-of-the-art within requirements handling.

This requirement handling system was first applied in the UEML project, and a variant of it is in day-to-day use by a commercial software vendor and its customers. We finally describe the overall experiences from use of the system.

1. Introduction

Anyone designing and developing products will confirm that high-quality requirements are key to success. This is also true for research projects. The ATHENA (Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Applications) integrated project (IP) is a research programme financed by the European Union within the Sixth Framework Programme (FP6). The focus of ATHENA [1] is *interoperability*, and to address this ever-apparent problem a diverse academic and industrial consortium with 19 partners is established.

ATHENA's prime objective is to be the most comprehensive and systematic European research initiative in the field of enterprise application interoperability, removing barriers to the exchange of information within and among organizations. It performs research and applies results in numerous industrial sectors, cultivating and promoting a networked business culture. To ensure that the research is heading in the right direction and solving real industrial headaches, collecting and prioritizing requirements is essential for the success of ATHENA.

A diverse group of industrial users, consisting of small and large companies in the telecommunications, aerospace, automotive, and furniture sectors, play a central and coordinating part in ATHENA. They provide requirements from practical experience and pilot cases. These requirements are converted to research issues to be solved by the research projects. These projects cover diverse fields such as Business Process Management, Enterprise Modeling, Service-Oriented Architectures, Model-Driven Architectures, and Ontologies.

2. ATHENA Requirements Engineering

In order to meet the evolving needs of all the different groups involved, ATHENA applies an Active Knowledge Modeling (AKM) [2] approach to requirements engineering. The Dynamic Requirements Definition System (DRDS) described in this paper consists of web interfaces and modeling environments that are customized, configured, and adapted through visual modeling.

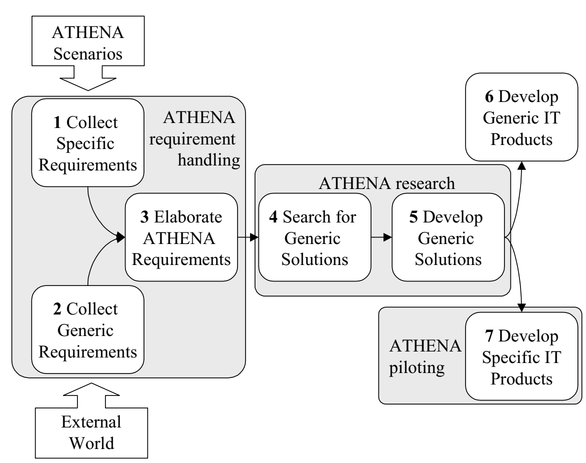

Figure 1. Overall requirement handling process.

Proceedings of the 2005 13th IEEE International Conference on Requirements Engineering (RE'05)
0-7695-2425-7/05 $20.00 © 2005 IEEE

The overall requirements handling process of ATHENA [3] is shown in Figure 1. Requirements are collected and organized in tasks 1, 2, and 3 by the requirements handling project, and then passed on to the research projects to find and develop solutions (tasks 4 and 5). Project partners and external software vendors may create IT products, while the ATHENA piloting project verifies solutions in the scenarios in which the requirements appeared, as well as other relevant scenarios (task 7). It is worth noting that the tasks are concurrent and this is an iterative process. The rest of this paper will describe how tasks 1, 2, and 3 are supported by modeling and visualization services.

2.1. Gathering Requirements

In ATHENA, requirements coming from the scenarios are called *specific* requirements since they relate to particular industrial scenarios (see Figure 2). Only requirements that are related to interoperability are relevant for ATHENA, and these are selected in task 1. Similarly, a set of *generic* requirements related to interoperability are gathered in task 2, coming from various external sources including previous research projects such as UEML [4].

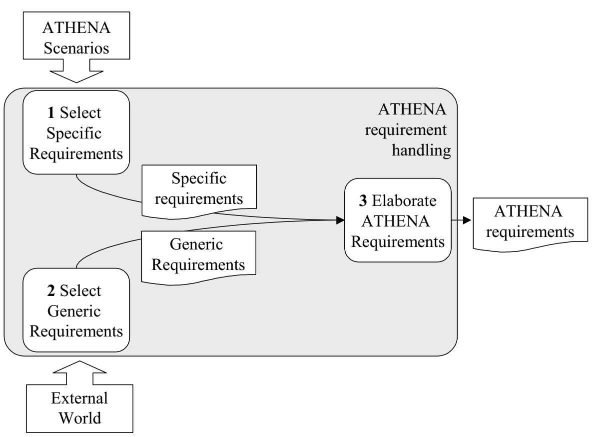

Figure 2. Requirements gathering.

2.2. Creating ATHENA Requirements

The specific and generic requirements are then elaborated and turned into ATHENA requirements. This is done by analyzing requirements in terms of the ATHENA objectives, drivers and pilot projects.

2.3. Classifying Requirements

To be able to more easily deal with similar and related requirements, they are grouped according to several classification schemes. A hierarchical classification method is applied. Each requirement may be related to as many classifications as are deemed relevant, and the set of classifications may be extended during the project.

The original classification structure is shown in Figure 3, but other orthogonal structures have also been defined for various sub-projects. A requirement may be classified according to any element in the tree-structure, so you may e.g. say that a requirement is classified as ‘business level’.

When creating ATHENA requirements from the specific and generic ones, it is possible to examine all specific and generic requirements that belong to a certain classification category.

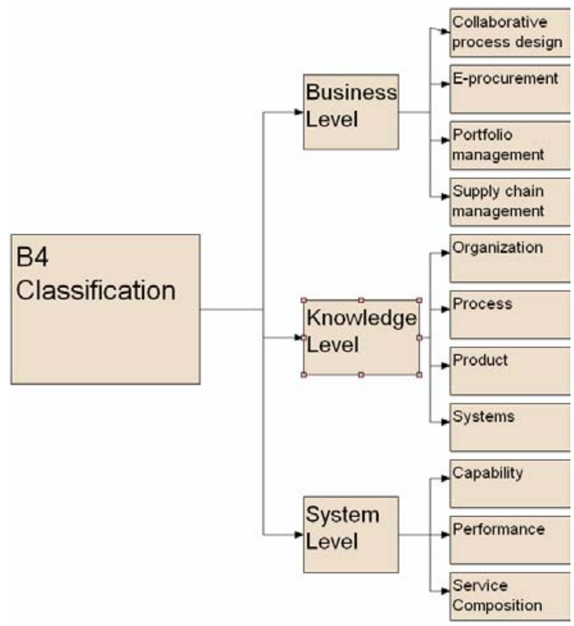

Figure 3. Classification example.

3. The ATHENA Dynamic Requirements Definition System (DRDS)

To support the requirement handling processes, an IT system with the components shown in Figure 4 was created. The starting point was a similar system that had already been used in the UEML project. The main parts of this system are a web user interface, a requirements database, and a visual model. The requirements database is shared between the web user interface and the modeling tool, meaning that users can seamlessly collaborate while using two completely different tools.

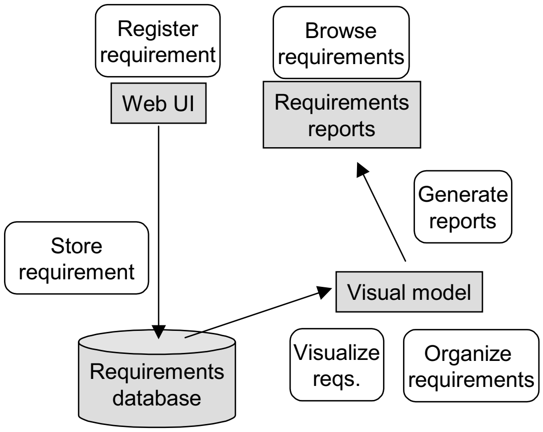

Figure 4. System architecture and processes.

3.1. Web User Interface

Most sporadic IT users are reluctant to install new software. Therefore, we decided that the requirement handling system had to have a web user interface, as shown in Figure 5.

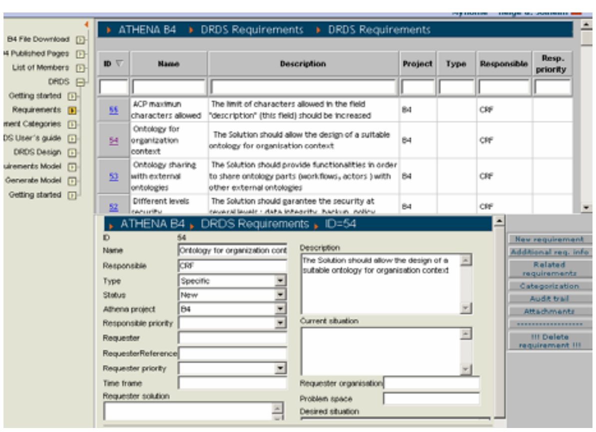

Figure 5. Main web user Interface.

The screen is divided into 3 main parts:

1. The left-hand menu allows the user to select function, such as "getting started", "requirements" or "manage categories". "Requirements" can be selected for viewing, editing and submitting new generic or specific requirements into the system.
2. The upper half shows the current list of requirements. This list can be filtered and sorted, e.g. to find existing requirements with given keywords. A requirement can be selected from this list.
3. Details of a selected requirement are shown on the lower half. Details of the requirement include its type, status (new, solved, rejected, etc.), who is responsible, originator person and organization.

A requirement may be related to any number of other requirements. For each pair of related requirements, it is necessary to select *how* they are related. The available options are duplicate, impacts, leads to, part of, and prerequisite for. In a similar manner, a requirement can be related to any number of categories.

There are also some administrative functions available in the web interface, such as administration of categories. Here the users can define which categories are to be available for classifying the requirements. Because the classification model is open for editing in this way, evolving and emerging user needs for new categories can be handled by the users themselves.

3.2. Requirements Database

A relational database is used for storing all the requirement information. This database has entities such as requirement, requirement type, status, ATHENA project, class, relationship, relationship type etc.

3.3. Visual Model

Storing a set of requirements and related information according to a predefined database schema is not a novel idea. However, when using a visual modeling tool that can be extended with any constructs necessary to deal with the requirements; we have a very different situation. Remember that the web user interface and the modeling user interface share the same data; the database (cf. Figure 4).

Metis [5] is a visual modeling tool that has pluggable metamodels. A metamodel defines the entities (e.g. objects and relationships) that can be modeled. This means that Metis may be used to model anything, as long as there exists a suitable metamodel. Anyone can create their own metamodel or extend existing ones to suit their purpose. This feature has been utilized in ATHENA to meet evolving and emerging requirements modeling needs. The wide and diverse audience, the concurrent and incremental structure of the requirements process, and the dynamic nature of the domain (research), implies that such ongoing adaptation is crucial for the success of the DRDS.

When visualizing something, there are numerous issues to resolve. We decided that each requirement should be represented by an object in the model. Furthermore, all requirements of each type (Generic, Specific, ATHENA or Other), are put together inside a container (cf. Figure 6).

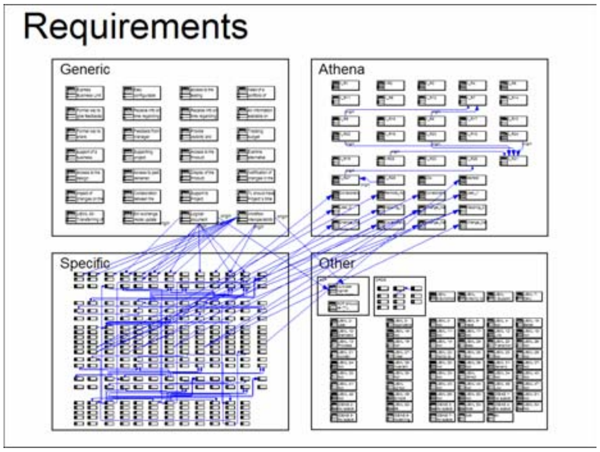

Figure 6. One container per requirement type.

The classes in the classification structure (Figure 3) are considered objects as well. They are shown in the lower left-hand part of the model in Figure 7.

The overall model structure was defined to have four main parts: requirements, classifications, attachments and reports. For each of the classifications, links are created to the corresponding requirements objects. An example of such a model is shown in Figure 7, (where the large number of links between specific requirements and classifications within the 4 different schemes are difficult to tell apart from one another).

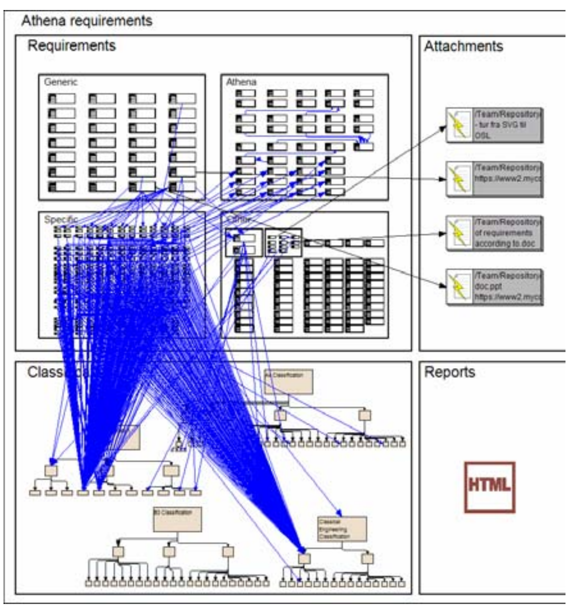

Figure 7. The whole requirements model.

Now, by selecting one requirement, you can see all its properties, as well as its links to other requirements, and its classifications. Similarly, it is possible to start at one classification and easily see (in the properties dialog) all related requirements. In Figure 8 three requirements related to the classification object "Portfolio management" are listed.

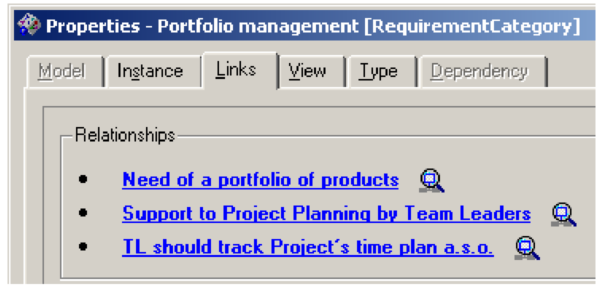

Figure 8. Links from a classification element.

A user of the modeling tool may also search and filter visually to show e.g. requirements related to a classification item. Figure 9 contains all requirements related to portfolio management.

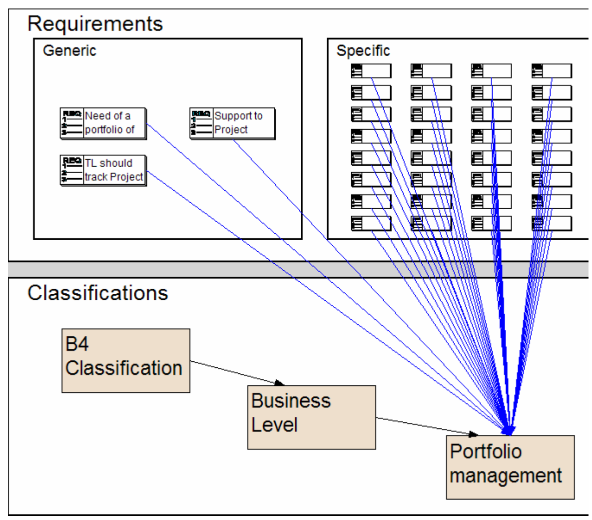

Figure 9. Highlighting requirements related to a classification item.

3.4 Requirements Reports

At the time of writing, which is in the first year of the ATHENA project, reports to the outside world are not emphasized. This is because few requirements have been fulfilled yet. However, it will soon be important to give feedback to external stakeholders to show that their requirements have been understood and categorized properly, as well as to show when the requirements are expected to be solved. Then we will need the capabilities of the modeling tool for performing what-if analyses, exporting and reporting selections of the data to other formats (documents, webs, xml, pictures etc.).

4. Extending the Requirements Model

Any ATHENA research project can extend the common requirements model and create its own augmenting models. Since the modeling tool allows for anything to be modeled, the research projects may model entities such as their solutions (in the form of standards, software, hardware, protocols, interfaces, APIs, etc.), their approach (projects, tasks, partners, architecture), or they may model scenarios to better describe the as-is situations in which the requirements arose. All or some of these entities may then be linked back to the common requirements model, showing which solution is expected to solve which requirement, when, how, and by who.

5. Usage Experiences

The DRDS was used with great success in the UEML project. Various users entered requirements in web-based user interfaces, and a requirements model was applied to organize the requirements according to a single classification structure in which each requirement was only allowed to have one classification.

In ATHENA we have limited experience with the system so far. However, we believe that connecting requirements to multiple classification structures will provide the various research projects with the overview they need to solve interoperability issues in the best possible manner without duplicating effort between them. As Figure 7 shows, 4 different classification schemes, representing different user groups with diverse viewpoints, have already been applied. When we started collecting requirements, only one of them (the one in Figure 3) was in place. The dynamic nature of the requirements system has thus been utilized.

The commercial software vendor Troux has used a variant of the DRDS system to collect and organize thousands of bugs and requirements for customers. In this system the main focus has been on communication with customers and giving feedback on priorities for next releases. Therefore, web-based reporting has been the hottest topic, though visual models were found to be valuable for internal communication. The system also implements a static requirements handling process, supporting five different roles (customer, tester, 1st line support, change management board, and developer). In a future version, we aim to utilize process enactment, so that the requirements handling process can be model-configured, customizable, and adaptable just like the web interfaces and the metamodels of Metis.

6. Summary and Conclusions

This paper has described how the ATHENA Integrated Project deals with requirements. Requirements are collected from industry or academia, as well as from specific pilot cases within ATHENA. These generic and specific requirements are consolidated into ATHENA requirements, and solutions to these requirements are proposed and researched. Once solutions are available, they are verified against the scenarios in which they originated.

A model-driven requirements handling system supports this entire process. A web-based user interface is available for the external world, and a powerful visual modeling tool is used for search, navigation, categorization, alignment, and prioritization of requirements. Both tools share the same underlying model, according to principles of Active Knowledge Models (AKM). This implies that no user interface or metamodel structure is hard-coded. Instead they are configured by visual models and evolve with emergent use of the tool. This flexibility has proved valuable for ATHENA. In our future work, we will seek to replace the waterfall model of software engineering with an incremental and continuous AKM approach to model-driven, user-composable platforms and solutions development (MUPS). In this approach, requirements will be integrated in the enterprise models that define the problem scenarios and configure the solutions.

Acknowledgements

We would like to thank the members of the ATHENA IP consortium, and in particular members of the Requirements project for their contributions to this work. The European Commission finances the ATHENA IP in support of the Strategic Objective “Networked businesses and government” in FP6.

References

[1] The ATHENA Integrated Project, http://www.athena-ip.org, 2004-2007.

[2] F. Lillehagen, “The foundation of the AKM Technology”, *Concurrent Engineering*, Jardim-Goncalves, Cha and Steiger-Garcão eds., Balkema publishers, 2003, pp. 700-715.

[3] ATHENA Consortium, “ATHENA Deliverable D.B4.1 Dynamic Requirements Definition Principles”, 2004.

[4] UEML Consortium, “Unified Enterprise Modeling Language”, http://www.ueml.org, 2003.

[5] Troux Technologies AS, “Metis”, http://www.trouxmetis.com, 2005.