

Model-Based Requirements Analysis with AutoRAID

Andreas Fleischmann, Eva Geisberger, Markus Pister, Bernhard Schätz
Fakultät für Informatik
Technische Universität München
Boltzmannstr. 3
85748 Garching, Germany
{fleischa|geisberg|pister|schaetz}@in.tum.de

Abstract

While software systems are increasingly developed by using model-based techniques, requirements engineering is most still performed text-based. Requirements are generally structured using text-patterns or linked outlines; as a consequence, only an indirect, weakly structured transition to design is possible. Model-based requirements engineering offers a solution to bridge this gap. Essential characteristics of model-based requirements engineering are a structured and integrated requirements model just as well as an effective, tool-based support for refinement and structuring from textual requirements to the design model.

1 Introduction

The core issue of model based requirements engineering is the introduction of a common product-model for requirements as well as design elements; and – based on this model - a step-wise structuring and classification of requirements in the development process (e.g. from unstructured text via structured text and tables to structured models of requirements). Thus a gradual, tool-based transition from requirements analysis to design is possible. The gap between analysis and design – with a textual document using only section/sub-section structure on the one side (as found, e.g., in DOORS [Tel04]), and (graphical) models with an elaborated structure using concepts like components, channels, states, or transitions on the other side (as found, e.g., in Simulink [MW02]) – is reduced.

The transition is performed by structuring the requirements step-by-step, guided by the model and enriching the model by design decisions during each step. Due to the increasing precision and modularization of the specification, a higher degree of analysis and generation techniques is available to improve the quality and efficiency of the development process.

In a nutshell, the model used to formalize requirements is similar to those practically applied models of requirements used manually for their classification, as found in approaches like the IEEE Guide for Developing System Requirement Specifications [IE98]. Thus, model based requirements engineering is integrated seamlessly in a well structured software development process: e.g., based on the model, tool-supported structuring and classification mechanisms can be offered to a domain expert, which are generally performed during an inspection or review process [DP04]. Besides supporting those structuring steps (e.g., identifying the different steps of a use case; identifying requirements concerning the interface of a system), analytical techniques commonly applied in an inspection can be mechanically applied to the model (e.g., each identified functionality must be broken down into sub-functionalities or allocated to a system component).

Obviously, the fundamental step of integrating of requirements into a model based development process consists of the integration of requirements into the product model. In the following, the integration techniques are presented that are implemented in the current version of the AutoRAID (AutoFocus Requirements Analysis Integrating Development) tool prototype:

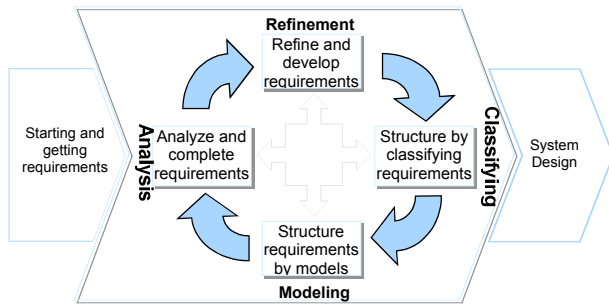


Figure 1: Activities in the Usage of AutoRAID

- *Structured requirements model*: Introducing domain-specific classes of requirements (e.g. architectural requirements, modal requirements, use case scenarios).
- *Connection to ModelViews*: Linking classes of requirements and classes of model elements (e.g. modal requirements to control states, architectural requirements to components).
- *Integration of textual requirements*: Integrating textual requirements into the model and offering pragmatic mechanisms for including textual documents into the development process.
- *Tool-based support for classifying and structuring*: Mechanising the step-wise structuring (e.g. identification of components in architectural requirements, of actuators in functional requirements) or generation (e.g. control states out of modal requirements).

The model views used in the approach presented here are taken from the specification tool AutoFOCUS [AF04], targeting the development of reactive systems. Based on these view, AutoRAID implements a first step in integration of requirements into a model-based development process.

Current work extends this first form of integration into a *strong integration of domain-specific requirements*. By using a product model with structured and domain-specific requirements and views, a stronger link between the requirements and model views in AutoFocus is supported.

2 The AutoRAID-Approach

The AutoRAID-approach consists of a systematic, iterative approach. The requirements are elaborated, structured, analysed and completed with help of domain-specific models.

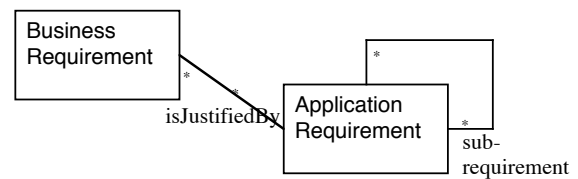


Figure 2: Refinement hierarchy

Requirements Engineering starts with the *elicitation* of requirements. Common techniques are structured workshops or interviews [SS98, KS98]. The integration of the results into a tool is important for a tool-based approach.

The iterative approach introduced in this paper consists of the activities *Refinement*, *Classification*, *Modelling* and *Analysis* (see figure 1). The requirements model is generated step by step. As well the behaviour of the system is defined and conditions (*Constraints*) for the further system definition (*System design*) and the implementation are elaborated.

The requirements are revised and completed until a consistent requirements- and system-specification is defined. This specification appropriately considers the goals and objectives of the involved stakeholders. This specification is the basis for the further development and revision. In the following chapters the activities of Figure 1 are explained more detailed.

2.1 Refinement and Development of Requirements

The requirements of product shall be defined according to achieve the objectives of the product. Thus a specification shall contain no requirements not supporting the objectives, because these requirements could cause additional and bootless costs for the product development.

Forward-Tracing [KS98] is used during the refinement of requirements in order to track these dependencies or constraints. The model of the refinement relationships used in AutoRAID is depicted in Figure 2.

A goal (*Business Requirement*) motivates or constraints the functionality, the design and the implementation of the software according to customer requirements and economical reasons. Thus it justifies specific properties of a product. A requirement (*Application Requirement*) describes the functionality and properties of a product or defines the interfaces to and the constraints for the design and the implementation respectively. These requirements can be vague and incomplete.

Requirements are refined and completed step by step. The refinement relationship is a hierarchy¹ that allows to back-trace the requirements to their goals. This

¹ directed, cycle-free graph

relationship between requirements and goals can be used for analysis and completion of the specification.

The specification and analysis of use cases and scenarios is a basic element of the elicitation and the structuring of requirements. Use cases are an expressive description of the workflows or tasks and thus of the application interface of the system.

2.2 Structuring of the Requirements with Classification

Requirements can be classified according to the following types in order to define the functionality of the application and the actors and components involved in the application process:

- *Use-Cases with scenarios*: Uses cases are application-processes described by exemplary representative activity-flows (scenarios).
- *Architectural requirements*: These requirements describe the structure of a system and its environment. The components and the communication channels can be documented with these requirements.
- *Modal requirements*: These requirements describe the operating modes of the application. Different states and transitions can be defined here.
- *Data type requirements*: These requirements describe data types that can be used for communication within the system and with the environment.

Use cases have a special purpose. By questioning the procedures, further requirements, especially system requirements, are derived and the functionality of the system is defined (cp. [HJ+98]).

The classification is the first step of creating a model and a system definition. The elements of the environment are described, the system boundaries are defined and the interfaces are specified.

These model elements can be specified further by linking more requirements to the defined model elements.

2.3 Structuring of the Requirements with Help of Models

The requirements will be connected to the formal models of AutoFocus step by step. There are two alternative sorts of linkage:

- Classified requirements (e.g. architectural requirements) describe the models, (e.g. structure diagram) and motivate elements as part of the model.

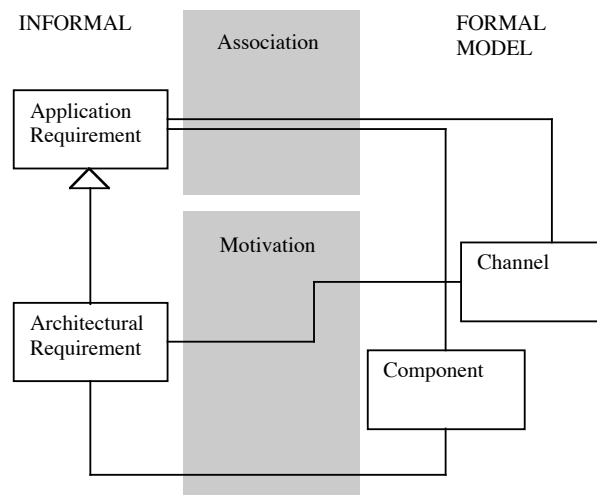


Figure 3: Requirements-linkage to Formal Architectural Models

Figure 3 depicts these linkages to model elements using the example of an architectural view.

These linkages of the requirements to formal models support further analysis and completion of the specification by verifying and validating the semantic of these AutoFocus models.

2.4 Analysis and Completion of Requirements

The objective of analysis is to point out inconsistencies and gaps in the specification and to evaluate requirements and system concepts in respect to product goals and customer requirements. The current structured specification is the basis for these analysis activities. The following methods are used in the AutoRAID approach:

- Inconsistencies can be detected according to the model relationships defined in AutoFocus.
- Specific questions to complete the specification can be asked according to the model relationships.
- Requirements and model elements can be put into a relationship according to the association between them and thus inconsistencies and incompleteness can be detected systematically and be revised.
- Usefulness of requirements can be questioned according to the refinement relationships.
- Problem formulation can be captured systematically by domain-specific questioning according to the classification

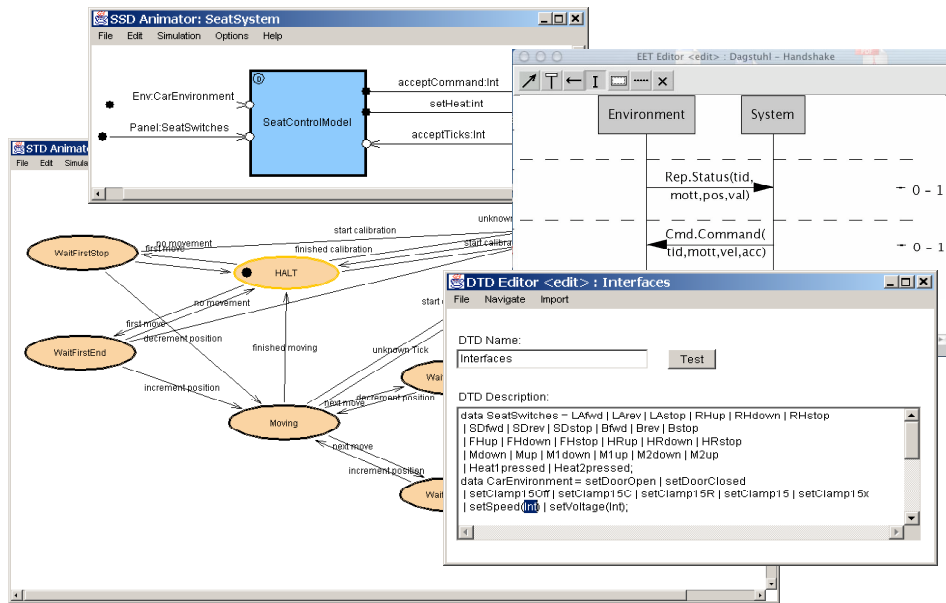


Figure 4: System Description in AutoFocus

The degree of structuring of the specification and the effectiveness of these analysis techniques increases with every iteration. Thus the specification is revised to be consistent and complete step by step.

3 Tool Support

With the development and the integration of the requirements management tool AutoRAID [AR04] into the tool AutoFocus [AF04], the concepts described in Section 2 are prototypically implemented.

AutoFocus is a model-based development tool for the design of embedded systems. It combines concepts of design-modelling, simulation and code and test case generation, and allows for the verification of software components. Figure 4 shows the mathematically founded system views of AutoFocus: hierarchical structure diagrams (SSDs), state transition diagrams (STDs), extended event traces (EETs), and data type definitions (DTDs).

Figure 5 shows the user interface of AutoRAID, which is integrated into AutoFocus. The project browser of AutoFocus (on the left hand side of figure 5 and 6) shows the tree of the requirements analysis (*Analysis*) in the upper part of the diagram; in the bottom part the design models of AutoFocus (*Component* „Car“) is displayed. On the right hand side there is the work area for the requirements and systems modelling. The subtrees of *Analysis* are *Source Contexts*, *Requirements*, *Use-Cases*, *Constraints* and *ModelViews*. These nodes represent the different steps in structuring and will be explained in the next chapters.

3.1 Generation and Import of Requirements

The tool AutoRAID allows to manually enter requirements or to copy and paste with import functions from structured documents (using an XML-interface) into a subtree of the requirements with the unique ID-number to be inserted. When doing that, the respective windows in the work area have to be filled with the requirements attributes like *title*, *description*, *patron*, *status*, *priority*, *rational* etc. (see figure 5). A more completed and structured HTML-Document can be generated from the elaborated and refined requirements in AutoRAID.

The events below the node *Source Contexts* define events by which a source document was generated (e.g. a meeting, telephone talks, conferences, etc.).

According to the refinement concept from chapter 2.1 the requirements are either stored as *Business* or *Application Requirements*. These can be subject to further refinement and hierarchical structuring in AutoRAID (compare for figure 5). Every *Application Requirement* must be derived from a *Business Requirement*.

3.2 Refinement and development of requirements

According to the general refinement principle any kind of *ApplicationRequirements* (even *UseCases*, *Constraints*) can be structured hierarchically. The possibilities in AutoRAID are:

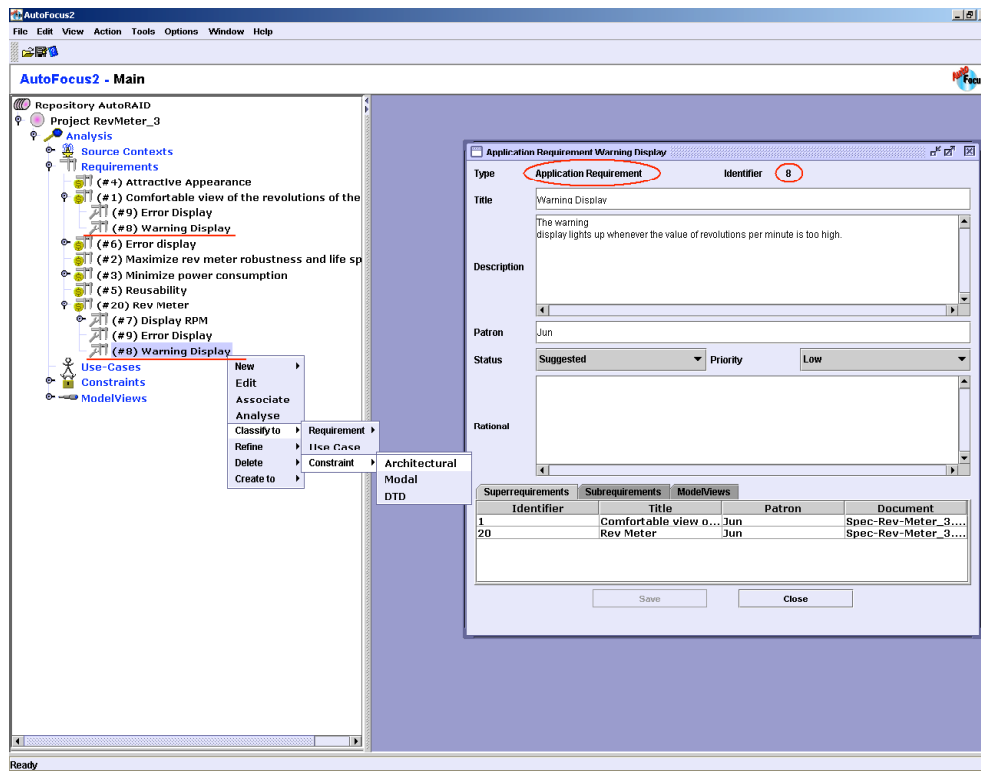


Figure 5: Project tree of AutoRAID with menu points and requirements attributes

- A *requirement* can be linked to one or more *requirements* (menu points „Refine“, „Copy“ or „Revise“).
- From an existing *ApplicationRequirement* a new *ApplicationRequirement* can be derived and linked to it as a subrequirement (menu point: “Create to”).

The created refinement relationships are inserted into the refinement attributes of the *requirements* automatically (list of the *Sub-* and *Superrequirements*, see figure 5). This graph-like refinement hierarchy can be changed by the user at any time (e.g. graphically by moving the elements within the navigation tree).

From the point of view of the methodology, the refinement and structuring of requirements are derived in most cases by the analysis of requirements according to use cases. With this kind of process analysis, the demand for functions, architectures, operating modes and data structures can be elaborated and be specified further (Modelling).

3.3 Classification and Modelling of Requirements

In AutoRAID *Application Requirements* can be classified according to *UseCases* (with the respective *Scenarios*) or

Constraints (*Architectural*, *Modal*, *Datatype (DTD)*). Figure 6 shows the definition of a use case in the tree on the left hand side and the respective scenario description of the „Basic-Flow“ in the right handed window.

Under the node *Requirements* the refinement hierarchy of all requirements is shown (figure 5). Under the nodes *Use Cases* and *Constraints* these are listed as a flat copy (figure 6).

Additionally to the function modelling, the structuring and refinement of requirements also allows to mark the demand for architecture, modes and data type requirements (*Constraints*) to be elaborated. With help of the menu item “Motivate” according model views can be defined in AutoRAID. This is the way how from architectural requirements as “Error Display” or “Warning Display” a component “Digital Display” can be generated in the tree *ModelViews* (figure 7). Simultaneously the tool generates an according component in the AutoFocus design tree (figure 6). These motivation relationships will be automatically listed in the appropriate motivations attributes of the *Constraints* and *ModelViews*. The definition of other model views and AutoFocus model elements is supported in a similar way (compare for figure 4 and 7).

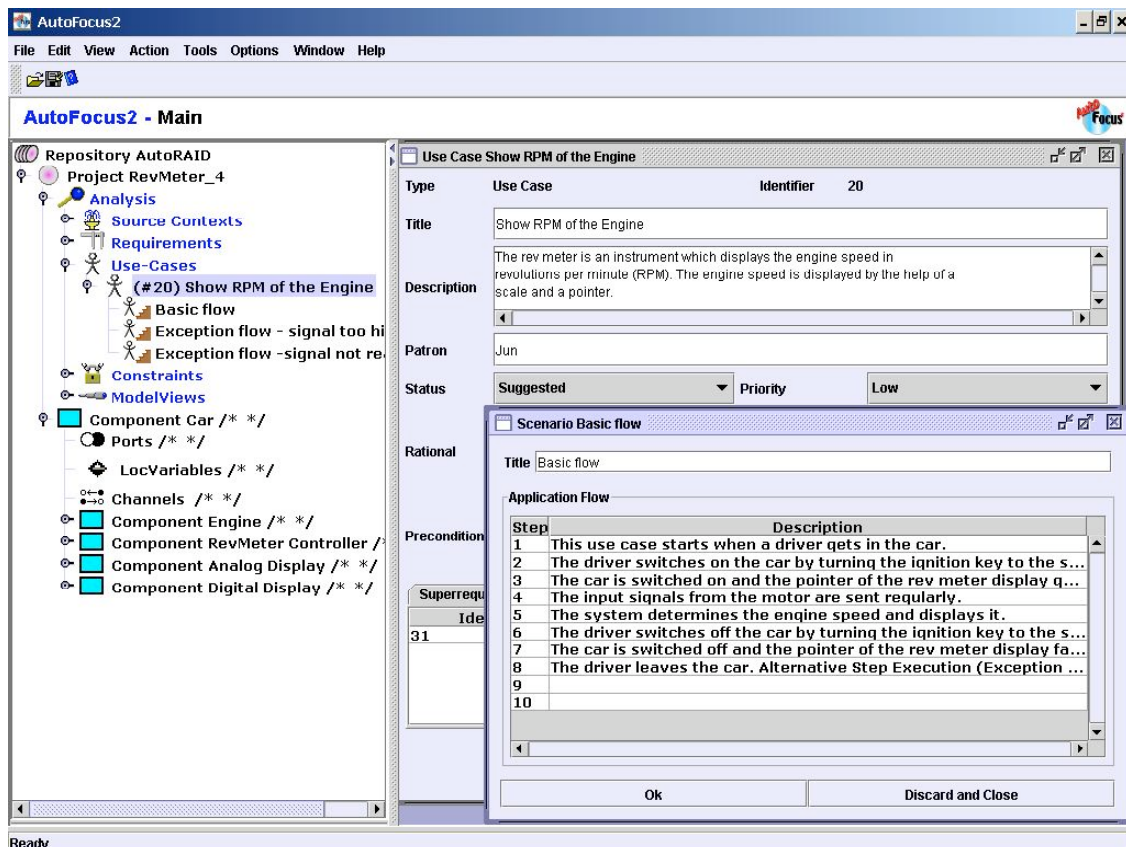


Figure 6: Specification of UseCase „Show RPM of the Engine“ with Scenario „Basic Flow“

As soon as the model elements are defined, these can be linked with any type of application requirements (compare for list of associated requirements in figure 7 (right hand side)).

3.4 Analysis and Completion

With the support of the basic model relationships in AutoFocus and the analysis of the structured and linked requirements, the requirement-models can be completed systematically step by step. This approach enables a gradual transition from the requirements analysis to the design. The integrated linkage between objectives, requirements and models allows for the validation and verification of requirements and systems designs.

Here, the AutoFOCUS framework supports the formulization of conceptual consistency properties, which are automatically checked by the tool [SP+02]. Thus, standard properties (e.g., coverage analysis to ensure that each requirement is covered by a constraint) can be checked. Due to the elaborate model of requirements, more sophisticated properties can be checked. (e.g., sub-constraints of a modal constraint are reflected in the sub-state relation of the corresponding state).

4 Related work

Core of the AutoRAID approach is the elicitation, structuring, analysis and validation/completion of the use and system requirements with the support of basic domain specific models as well as the consistent derivation of the requirements and system definitions from goals (forward and backward tracing). By the realization of this concept in the tool AutoRAID and its integration into the mathematically sound specification tool AutoFocus the verification possibilities can be used for the validation and completion of the requirements.

Approaches to the structuring of requirements with the help of functional models can be found in the work of Kotonya and Sommerville about VORD [KS95], Leite and Freeman („Requirements Validation Through Viewpoint Resolution“) [LF01] as well as Lamsweerde et al. about KAOS [Lam01].

Core of VORD is the analysis and structuring of requirements according to the usage point of view – according to “services“. The services of the system are described by scenarios and non-functional requirements are assigned to these services. In turn these services can be

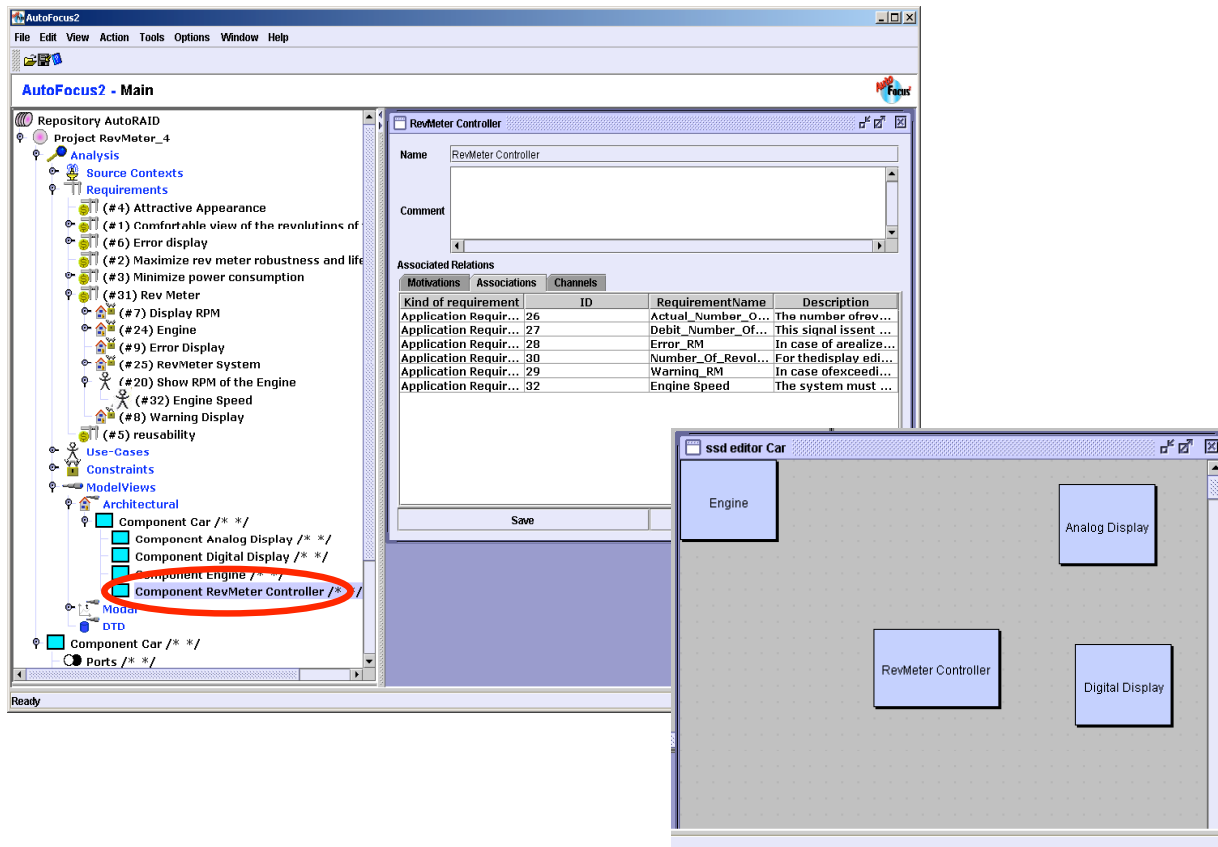


Figure 7: Definition of ModelViews and Generation of AutoFocus Models

specified by means of “event traces” and state transition diagrams. The tool support of VORD allows to recognize conflicts between requirements of different operational viewpoints (service specifications) systematically and to resolve these. Till now, a validation of the requirements by assigning them to mathematically sound models and a corresponding verification tool isn’t given in VORD.

Also Leite and Freeman structure requirements of different view points according to basic models – data view, “Actor”-/Process view, architectural view – and recommend corresponding heuristics, with which conflicts between these requirements can be discovered and revised. A mapping of this very general structuring concept to precisely or mathematically sound statements with corresponding verification possibilities isn’t given here either.

Goal-oriented approaches like KAOS [Lam01] analyse and work out requirements (Goals) with the help of HOW- (top-down) and WHY-questions (bottom-up). In addition KAOS defines a methodical concept for the refinement of goals and the mapping of the won detailed requirements to software components (agents). This way a structuring concept of requirements according to functional (Goals) and non-functional requirements (Softgoals) as well as the consideration of the requirements regarding its relations under each other is

suggested. As soon as the requirements on agents are derived, they can precisely be specified and verified with the help of temporal logic. A stepwise acquirement and structuring of the requirements with the help of functional models is not given in KAOS since there is a gap between the methodical structuring of the Softgoals and the functional structuring of the detailed requirements on agents.

In [DP04] a review-based approach is described which structures textual requirements by means of tabular Use Case descriptions as well as state-chart diagrams. AutoRAID simplifies such a review process by the tool based support of single transformation steps and further offers analysis methods as well as the inclusion of further aspects like data and structure.

5 Summary and Outlook

AutoRAID supports model-based requirements analysis by means of a structured and stepwise transition from textual requirements to the design model. Core functionalities are a detailed conceptual model with different classes of requirements (e.g. architectural constraints, modal constraints) and tool supported steps to detail and elaborate (e.g. classification, model generation). Current work extends the model to a *deeper integration of domain-specific requirements*. By use of a product model with enhanced domain-specific requirements (e.g. timing requirements) and views (e.g., deployment architecture), enhanced structuring and increased linking between the requirements and model views are supported in the AutoFocus framework. Objectives here include support for complex analytical techniques (e.g. consistency of the interfaces of the architectural constraints and the interfaces in the functional requirements), along with detailed generative steps (e.g. generation of sequence diagrams based on structured use case scenarios).

Acknowledgement

We thank the students of the AutoRAID project team for their commitment to the project. We are furthermore grateful for the support of Tobias Hain and Jewgenij Botaschanjan in implementing the AutoRAID tool prototype, as well as Frank Stöckel and Rupert Wiebel for their support during the design of AutoRAID. We finally thank Sabine Wildgruber for her support in preparing this contribution.

Bibliography

- [AR04] Website of AutoRAID, including documentation, screenshots and download
<http://wwwbroy.in.tum.de/~autoraid/>
- [AF04] Website of AutoFOCUS including documentation, screenshots, tutorials, download
<http://autofocus.in.tum.de>
- [DP04] Denger, C. Paech, B. "An Integrated Quality Assurance Approach for Use Case Based Requirements." In: Rumpe, B. Hesse, W. (eds). Modellierung 2004. Springer, LNI, 2004.
- [MW02] The MathWorks Inc. Simulink – Model-Based and System-Based Design – Using Simulink, 2002.
- [HJ+98] Haumer, P., Jarke, M., Pohl, K., Weidenhaupt, K., Scenarios in System Development: Current Practice, IEEE Software, Vol. 15, April 1998, pp. 34-45
- [IE98] Software Engineering Committee of the IEEE Computer Society. IEEE Guide for Developing System Requirements Specifications. IEEE Standard 1233-1998. IEEE Computer Society.
- [KS98] Kontonya, G. and Sommerville, I., „Requirements Engineering: Processes and Techniques“, John Wiley & Sons, 1998.
- [Lam01] van Lamsweerde, A., „Goal-Oriented Requirements Engineering: A Guided Tour“, Invited Paper for RE'01 - 5th IEEE International Symposium on Requirements Engineering, Toronto, August, 2001, pp. 249-263.
- [SS98] Sommerville, I. and Sawyer, „Viewpoints for requirements elicitation: a practical approach“, Proc. IEEE Int. Conf. on Requirements Engineering, Colorado Springs. IEEE Press, P. 1998.
- [KS95] Kotonya, G. and Sommerville, I.: “Requirements Engineering With Viewpoints”, Technical Report, Lancaster University, 1995.
- [SP+02] Schätz, B. Pretschner, A. Huber, F. Phillips, J. Model-Based Development of Embedded Systems. In: Briel, J.-M.; Bellahsene, Z. (Eds.): OOIS 2002 Workshops, Springer LNCS, 2002
- [Tel04] Website of Telelogic including DOORS
[http:// www.telelogic.com](http://www.telelogic.com)