Tool Support for Scenario-Based Architecture Evaluation

Steffen Thiel, Andreas Hein, and Heiner Engelhardt

Robert Bosch Corporation
Corporate Research and Development, Software Technology (FV/SLD)
Eschborner Landstr. 130-132, D-60489 Frankfurt, Germany
+49 (69) 7909 – { 518 | 512 | 501 }
{ steffen.thiel | andreas.hein1 | heiner.engelhardt }@de.bosch.com

Abstract

Architectural considerations play a key role in the success of any software-based development project. Architecture evaluation is an early risk reduction method for identifying risks that prevent a system or product line meeting the organization's business goals and customer needs. This paper introduces a tool that supports architecture evaluation. It gives an overview on its information management capabilities and discusses development issues as well as the underlying data model.

1. Introduction

Architectural considerations play a key role in the success of any software-based development project. Architecture evaluation is an early risk reduction method for determining whether the system or product line will satisfy the desired business and quality requirements. An important prerequisite to achieve this is getting an understanding of the consequences of architectural decisions with respect to those requirements.

Unfortunately, requirements specifications are often not definitive enough in practice, neither for architectural design nor for evaluation. As a consequence, requirements must be made explicit in order to be useful for development. Scenarios are practical in this respect since they allow to describe concrete interactions between the stakeholders and the system. They are, for example, useful in understanding run-time qualities such as performance and reliability. This is because scenarios specify the kinds of operations over which the qualities need to be measured, and the kinds of failures the system will have to withstand. Therefore, scenario-based methods have been proven useful in practice for evaluating architectures during a review [1, 3].

This paper describes a tool that supports scenario-based architecture evaluation. In Chapter 2, we sketch basic steps performed during an evaluation. Chapter 3 introduces AET, a tool developed by the authors that supports the evaluation team during a review. In Chapter 4 development issues of the tool are discussed. Finally, Chapter 5 concludes with a short summary and gives an outlook on further development activities.

2. Architecture Evaluation

The goal of an architecture evaluation is to identify risks that prevent the system or product line to be successful. Successful systems meet the organization's business goals and satisfy the customer needs.

The Software Technology department of Robert Bosch Corporate Research and Development performs architecture evaluations for business units [2, 3]. Some of these evaluations are based on [1]. Typical activities of [1] are the following:

Step	Description
Presentation	
1	*Present method:* The evaluation team describes the evaluation method to the assembled stakeholders (typically, architects, managers, marketing, integrators, testers etc.).
2	*Present business drivers:* The marketing representative describes what business goals are motivating the development effort and hence what will be the primary architectural drivers (e.g., high availability or high security or time-to-market).
3	*Present architecture:* The architect describes the proposed architecture, focussing on how it addresses the business drivers.
Investigation and Analysis	
4	*Identify architectural solutions:* Determine the central mechanisms (e.g., architectural styles or patterns) used in the architecture.

5	*Brainstorm and prioritize scenarios:* The stakeholders elicit scenarios to make business drivers and important requirements more concrete. The scenarios are then prioritized according to a ranking scheme (e.g., market importance and effort/cost).
6	*Analyze architecture:* Evaluate the architectural decisions made to achieve the high-priority-scenarios. This is supported by examining the architectural solutions from step 4 and by identifying those design elements that are affected by the scenarios.
Reporting	
7	*Present results:* Present the findings (e.g., risks) to the audience and summarize them in a written report.

In the past, the results of an evaluation have been documented in prose. As a consequence, the access on the results of those evaluations was quite inefficient and unsatisfactory. This motivated us to start the development of a database application. One goal was to create an experience repository of architecture evaluations, another to speed up information access and report generation. In the next chapter we describe the current status of the tool.

3. Architecture Evaluation Tool

The Architecture Evaluation Tool (AET) is a research tool developed by the authors at the Software Technology department of Robert Bosch. It supports a review team in documenting results and managing information during an architecture evaluation. AET makes reporting more convenient and allows exploring the content of an evaluation.

AET uses two different databases to store information: one for general data and one for project data. The general database contains static data – this means, data that does not depend on a specific system context. General data such as general analysis questions or scenarios can then be used to support the evaluation of a particular system.

The project database contains project-specific information obtained during an evaluation (dynamic data). It includes data such as qualities, scenarios, architectural approaches, and risks that have been identified and examined during evaluation.

In the following we describe how AET can be applied in practice during an architecture evaluation.

3.1. Requirements and Qualities

During the presentation of business drivers and the architecture (step 2 and 3), a lot of information about requirements and quality attributes is usually obtained from stakeholders. This information can be recorded in AET for later exploration and reference. Business goals, functional requirements, and design constraints can be put into a requirements list. Quality attributes can be documented in a quality tree. A sample quality tree for a fictitious Embedded Vehicle Control System (EVCS) is shown in Figure 1.

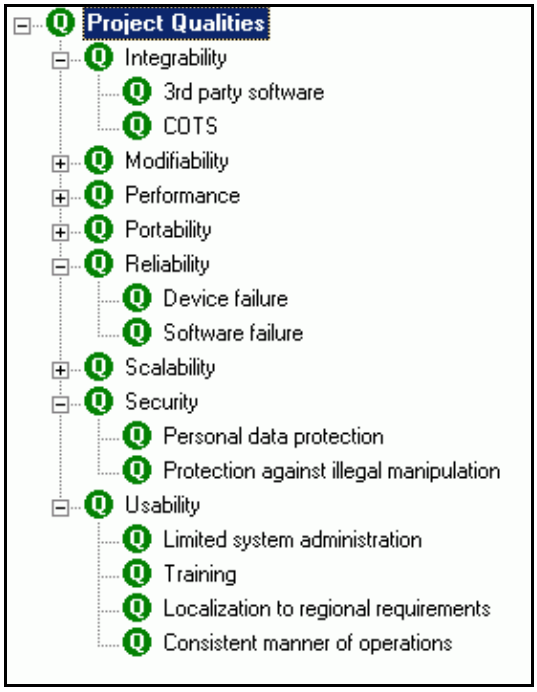

Figure 1. Quality Attribute Tree

Optionally, a starter set of typical quality requirements and scenarios for the type of systems under evaluation (e.g., embedded automotive systems) can be generated from the general database.

Each quality attribute may contain one or more sub-factors, as shown in Figure 1. Sub-factors describe specific stakeholder concerns of the quality. For example, in Figure 1 “personal data protection” is of specific concern for security. Note that each item in the quality tree can be moved easily. This allows quick modifications during an architecture evaluation.

3.2. Scenarios and Prioritization

AET allows to record the scenarios gathered in step 5 in a scenario list, as illustrated in Figure 2. Scenarios can also be described in more detail. For example, you can document potential stimuli and responses in order to make the expected behavior more concrete. In addition, you can link scenarios to a particular quality attribute or business goal in order to document that it contributes to that attribute or goal. In Figure 2, the selected scenario

contributes to “quick start up” which is a sub-factor of performance.

Furthermore, you can assign stakeholder priorities to each scenario, as defined in step 5. The scenario priorities then drive the further analysis. In the example of Figure 2, we use two dimensions (business importance and architectural difficulty) and three values (High, Medium, Low) for prioritization.

However, AET allows to adapt the dimensions and priority scale to fit individual needs of the evaluation, as shown in the lower right of Figure 2. Scenarios can easily be sorted according their priority such that the most important ones (high importance, high difficulty) appear at the top of the list.

3.3. Scenario Analysis

Each scenario can be analyzed in AET. Usually, you start with the most critical scenarios. There is room for a detailed description of the analysis results, including text and pictures. For example, you can describe the architectural elements that contribute to a particular scenario. You may also like to document how the architecture would need to be changed to accommodate a scenario. The description is stored in HTML-format in the database for post-processing.

Furthermore, AET allows to classify important findings of the analysis. Important finding are, for example, risks or tradeoff points. Risks arise from architecturally important decisions that have not been made, yet. A tradeoff point occurs when multiple quality attributes are differently affected when changing one architectural parameter. For example, improving throughput may result in reduced reliability.

When recording a finding, AET directly links the finding to the scenario under analysis. This is very useful since it allows you to easily trace back risks, tradeoffs, issues etc. to its source – the scenario. You may follow the trace to obtain a more detailed description of the analysis. Storing traces also supports statistics, for example, about which scenarios are most critical for the success of the system.

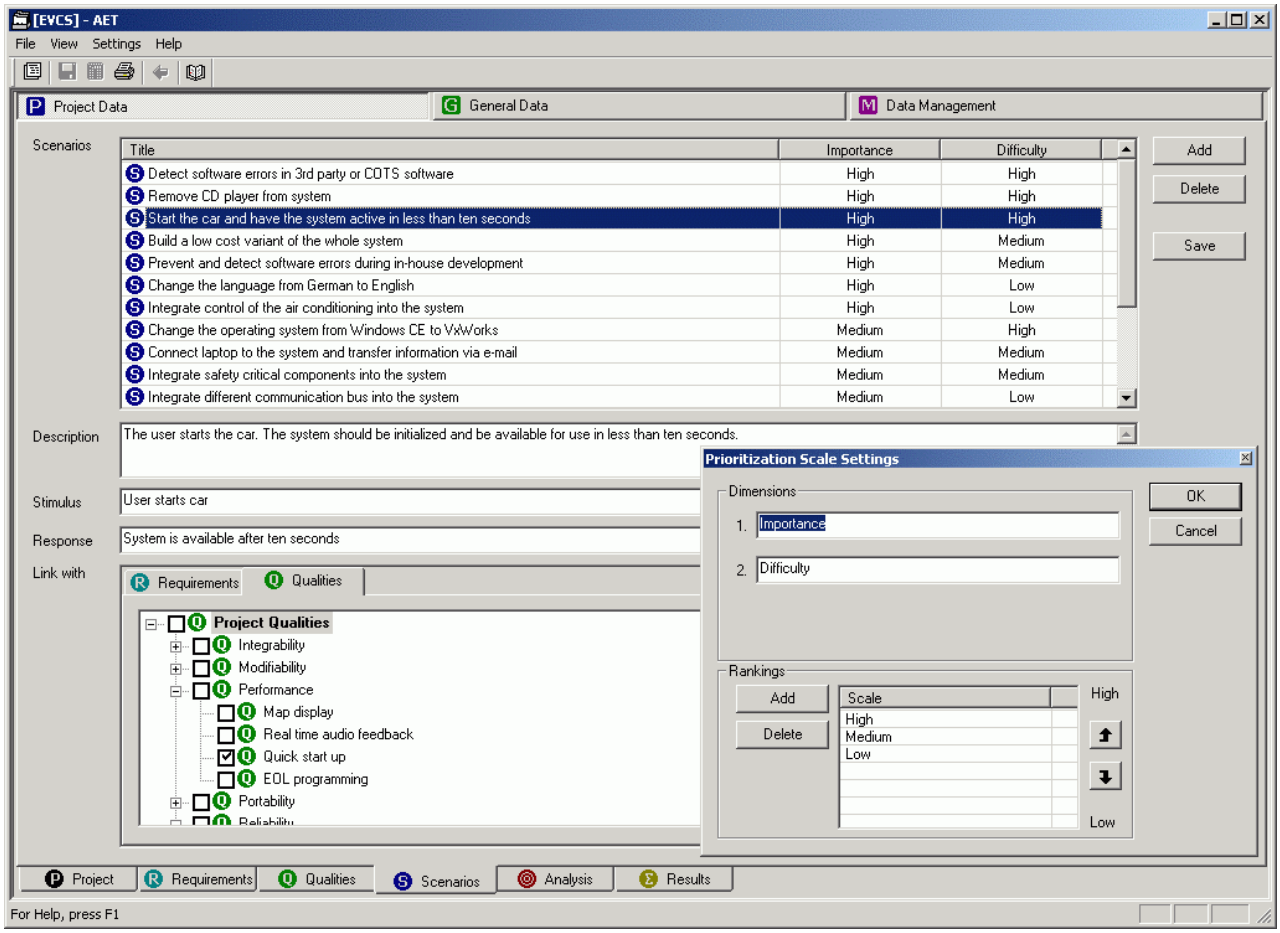

Figure 2. Classification of Scenarios in AET

3.4. Analysis Results

Since the number of risks identified during an evaluation can be high, AET allows to classify them in risk themes [1]. Risk themes summarize key architectural issues that pose potential future problems for the success of the system. For each risk theme, AET allows to assign one or more findings. In addition there is room for a detailed discussion of the risk theme. Risks and risk themes can be clearly arranged in a "result tree," as shown in the lower part of Figure 3. This tree is automatically generated by AET based on the relationships between findings and risk themes.

AET can also generate a "utility tree," as illustrated in the upper part of Figure 3. This tree represents a summary of the elicited scenarios and priorities together with the respective sub-factors and quality attributes. As shown in Figure 3, four scenarios have been documented to address the modifiability concern of supporting "multiple customers."

Finally, the results documented in the AET project database can be included in a written report. The different tree views visualize the results of the evaluation in a concise form. This supports clarity and understandability of the documentation.

4. AET Development and Database Model

AET is an easy-to-use application. It is implemented in C++ and runs on Microsoft® Windows operating systems. It uses a commercial database system for storing and retrieving data which has reduced the development effort drastically. Furthermore, AET deals efficiently with its resources. It is thus a suitable companion for a mobile application at the customer site.

From the architectural perspective, AET is organized in three layers: presentation, application, and data management. The presentation layer is responsible for user interaction and data presentation. Data post-processing such as scenario sorting or combining data from different database tables is done in the application layer. The data management layer provides low-level services to access and maintain the database.

Figure 4 shows a simplified model of the project database. For each evaluation project you can record individual requirements, scenarios, architectural decisions, findings, and risk themes. Priority dimensions and scales are global to a project. Each scenario can have a ranking. The ranking must conform to the global scheme defined for the project.

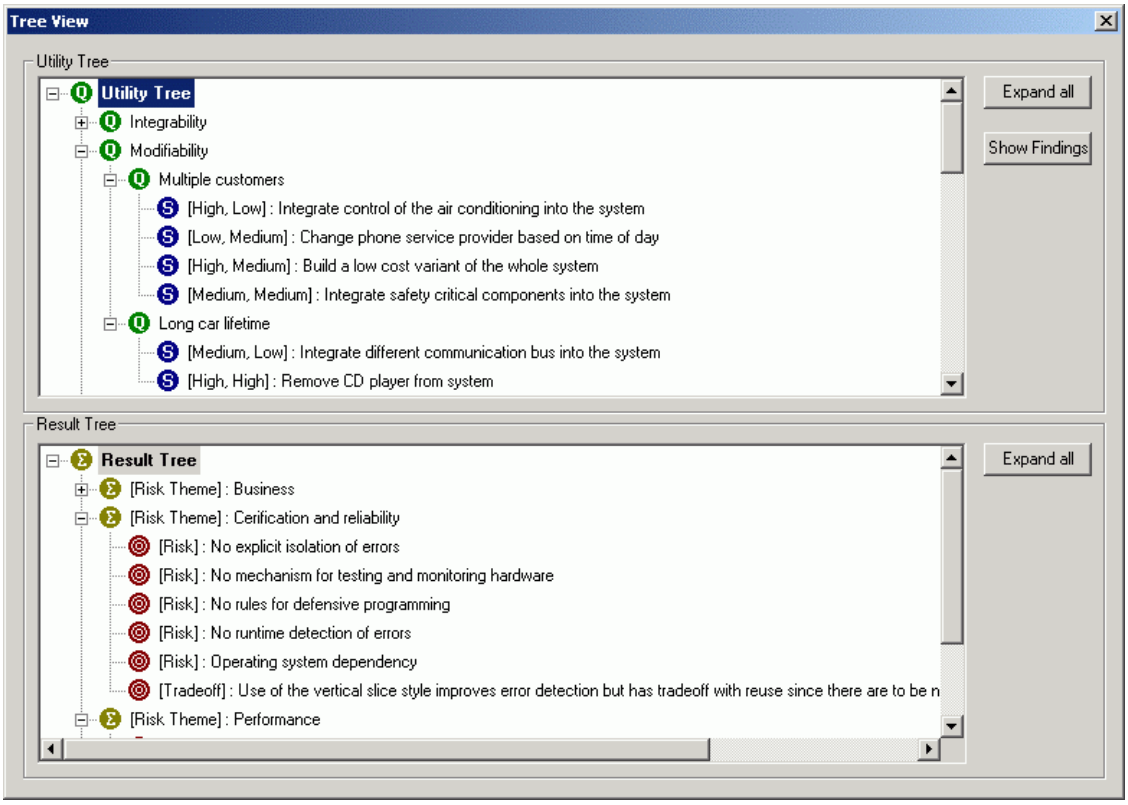

Figure 3. Tree View of Qualities, Scenarios, and Analysis Results

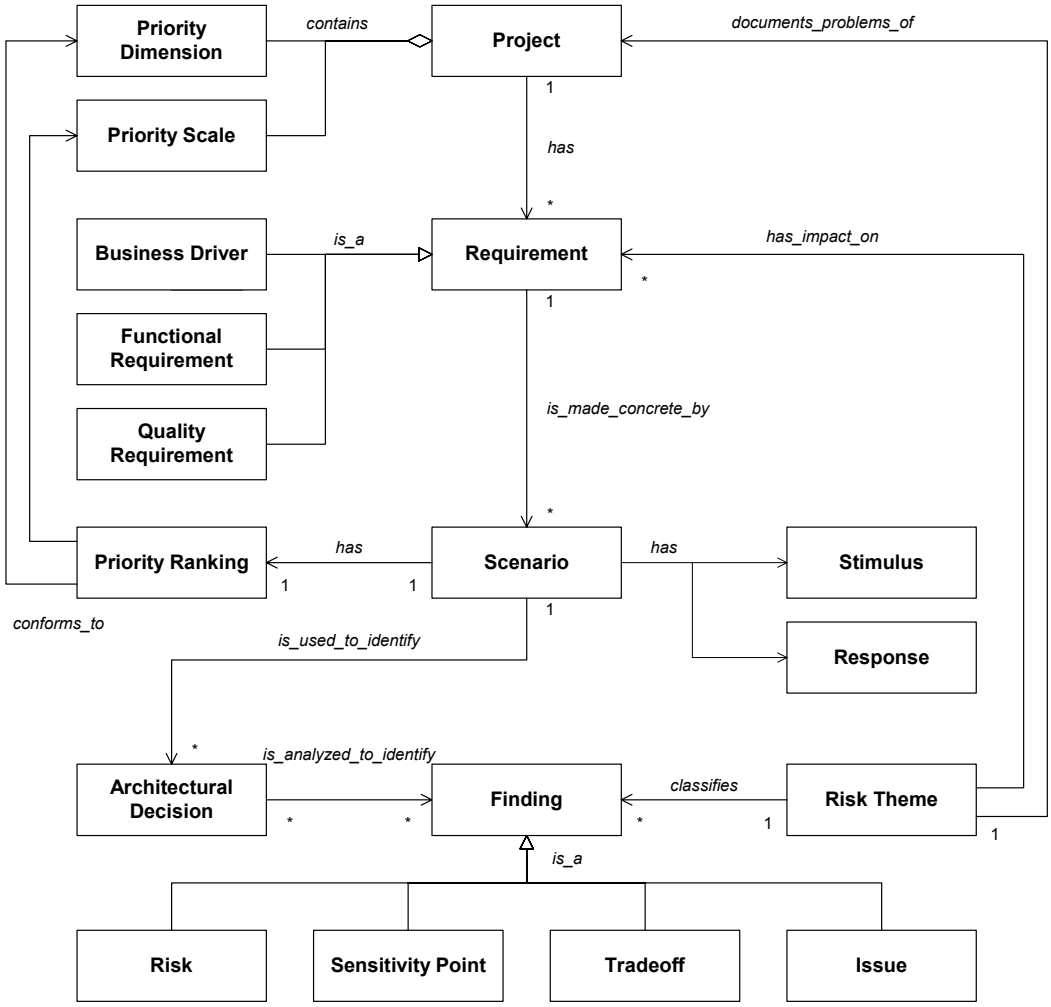

Figure 4. AET Data Model

Scenarios have a stimulus and a response. They are explored in order to identify architectural decisions. These decisions can further be analyzed to identify particular findings. A finding can be a risk, sensitivity point, tradeoff, or issue. Risk themes classify and summarize findings. Each risk theme has an impact on one or more requirements – this means, some of the business drivers or qualities cannot completely be met. The risk themes document the problem areas associated with the system under evaluation. They indicate how close an organization is to fielding a successful system.

5. Summary and Outlook

In this paper we have introduced AET, a tool that supports scenario-based architecture evaluation. We first discussed typical steps of an architecture evaluation. Next, we gave an overview of AET, and, finally, we sketched development issues and the AET data model.

AET is still under development. We plan to improve the export interface for report generation and to include functionality for querying the project database and for performing evaluation statistics.

6. References

[1] P. Clements, R. Kazman, M. Klein: **Evaluating Software Architectures: Methods and Case Studies**; Addison-Wesley, 2002.

[2] S. Ferber, P. Heidl, P. Lutz: **Reviewing Product Line Architectures: Experience Report of ATAM in an Automotive Context**; Proceedings of *4th International Workshop on Product Family Engineering (PFE-4),* pp. 364-382, Bilbao, Spain, October 3-5, 2001.

[3] S. Thiel: **On the Definition of a Framework for an Architecting Process Supporting Product Family Development**; Proceedings of *4th International Workshop on Product Family Engineering (PFE-4),* pp. 125-142, Bilbao, Spain, October 3-5, 2001.