Abstract

Researchers from requirements engineering and software architecture had emphasized the importance of Non-Functional Requirements and their influence in the architectural design process. To improve this process we have designed a tool, ArchiTech, which aims to support architects during the design process by suggesting alternative architectural decisions that can improve some types of non-functional requirements in a particular project, and facilitate the reuse of architectural knowledge shared between projects of the same architectural domain (e.g., web-based applications).

1 Introduction

Non-Functional Requirements (NFRs) are one of the main targets of research in the Requirements Engineering community [1]. A trending topic along this line is the analysis of relationships between NFRs and software architectures (e.g., [2]). From the perspective of the software architecture community, many researchers have stated that architectural decisions are the core of software architecture [3], and how these decisions can be reused among projects by having a common knowledge base, normally referred as Architectural Knowledge (AK) [4]. Therefore, it seems natural to explore the links among NFRs, architectural decisions and AK and to look for tool support for their coordinated management.

In this paper we present ArchiTech, a tool to guide architects in the architectural decision-making process. ArchiTech integrates two subsystems: an Architectural Knowledge (AK) manager, ArchiTech-CRUD, and an architectural decision-making-assistant, ArchiTech-DM (see Figure 1).
2 ArchiTech Description

The ArchiTech tool is part of a bigger envisioned framework to deal with NFRs in Model-Driven Development [5].

ArchiTech starts from a set of quality requirements and constraints derived from a SRS by the architect. As result of the decision-making process (described in Section 2.2), ArchiTech will provide a set of architectural decisions and an overall evaluation of the expected quality as consequence of applying the resulting decisions. The results obtained from ArchiTech-DM require the management and maintainability of Architectural Knowledge, these facilities are provided by Architech-CRUD in form of interrelated CRUD (Create-Read-Update-Delete) operations (described in Section 2.1) applied to the concepts defined in an AK ontology [6].

2.1 ArchiTech-CRUD

This subsystem provides a graphical user interface for the domain expert to operate with the AK. The CRUD operations are specialized for four different types of knowledge:

- **Architectural element.** The domain expert has to define the elements (e.g.,
architectural styles -SOA, layered, etc., components -services, packages, etc., technologies -DBMS, RESTful vs. W3C, etc.-) that are to be used to structure any software architecture. We use four architectural views (logical, deployment, development, and platform) to classify these elements.

- **Properties.** Each architectural element may have values for one or more properties that are defined by the domain expert (e.g., the property License may be used to classify and reason about OSS technologies).

- **Types of NFRs.** We give freedom to the domain expert to define (and reuse in multiple projects) the most appropriate quality model for her interests (e.g., the S-Cube quality model to design SOA systems [7]).

- **Architectural decisions.** The domain expert has to define the decisions that are more habitual in a particular architectural domain (e.g., web-based system, service-based system, etc.), and which types of NFRs are affected by each alternative. Decisions can be higher-level (e.g., which architectural style to apply) or lower-level (e.g., which DBMS to choose).

In order to provide management facilities, the AK must be persistent and easy to share among projects. To this end we provide an embedded database, and we have also added an option to export the stored AK to an XML file.

### 2.2 ArchiTech-DM

This subsystem uses our method, called Quark, to guide software architects in NFR-driven decision-making. Quark starts from the SRS (from which we focus in NFR at this stage), and ends with a set of architectural decisions and the overall evaluation of the software quality. The Quark method delivers an iterative process divided into four activities:

1. **Architectural Specification.** The architect specifies the QRs and constraints (using the SRS as basis) that will drive the architecture decision making. For example, a QR could be “performance should be high” (in other words, more a goal than a requirement), and constraints could be “the database management system (DBMS) must be MySQL 5”. As happens in this example, QRs may be at a high level of abstraction.

2. **Decision Inference.** The ArchiTech tool uses the AK to generate a prioritized list of decisions (e.g., the decisions that satisfy more constraints and better comply with the stated QRs) using simulated annealing [8].

3. **Decision Making.** The architect decides what decisions are to be applied from the ones generated in the previous activity. When the architect makes a decision, some issues may rise (e.g., we may be selecting the “data replication” decision, but we could have already selected a DBMS not supporting this feature).
Architectural Refinement. The ArchiTech tool identifies possible issues and suggest actions that to resolve them (e.g., following the previous example, the tool may suggest to use a DBMS with data replication).

After the fourth activity, we may end the process by accepting the resulting set of architectural decisions or use the suggested actions provided by the tool and start a new iteration. Once a new iteration starts, the architect is free to change any of the previously defined requirements (e.g., s/he may want to soften some of them to get more alternatives).

3 Conclusion

In this paper we have described the principal parts of ArchiTech and summarized the specific use cases of the tool.

A detailed description of the system may be found at (www.upc.edu/gessi/architech/index.html), where the current version is available for download.
Also, the details of the design of ArchiTech (architectural description, technologies used, etc.) are available in [9]. Finally, a screen-shot of the tool is shown in Figure 2.

Acknowledgment
This work has been partially supported by the Spanish MICINN project TIN2010-19130-C02-01.

References