The Requirements Editor RED

Störrle, Harald; Kucharek, Maciej

Published in:
Proceedings of the joint track "Tools", "Demos", and "Posters" of ECOOP, ECSA, and ECMFA, 2013

Publication date:
2014

Document Version
Peer reviewed version

The Requirements Editor RED

Harald Störrle, Maciej Kucharek

Department for Applied Mathematics and Computer Science
Technical University of Denmark
hsto@dtu.dk, kucharek.maciej@gmail.com

1 Motivation

The Requirements Editor (RED) has been conceived as a tool to support teaching a major Requirements Engineering class at the Technical University of Denmark (DTU). The course covers a wide variety of techniques in a hands-on fashion, from stakeholder analysis and goal modeling via interaction design and classic textual requirements to UML models. The need of tool support is quite obvious, but all the tools on the market covered only a small segment of these techniques, used a different (and often inconsistent) terminology, and were hard to customize. After several failed attempts to use pre-existing tools, we decided to build our own. Since this was for a RE course, we did a thorough requirements analysis up front, using the techniques taught in the course.

2 Goals, Constraints, Requirements

The primary goal of this project clearly is to support the course, in particular, to help the students understand the course material. This would, indirectly, help the teacher deliver a better course, and thus help both main stakeholders equally. As a consequence, we demand that **R1** all major topics of the course are covered, **R2**that terminology is consistent with the course material, and **R3** that students should be be freed from mechanical tasks, leaving more time for the actual course contents. Furthermore, since this is a tool that is supposed to be used by up to 70 students each year, it was necessary that **R4** the tool would run on all major platforms, with high degrees of **R5** robustness and **R6** usability. Finally, it was clear on the outset that not all desirable features would be implementable in a single increment, but instead, that many different students would be working on it over a prolonged period of time, so that **R7** maintainability was an important quality.

3 Project History

Faced with these requirements, we decided to create RED using the Eclipse Rich Client Platform (ERCP). At the time, Eclipse 3 (Indigo) was the most recent version, so that was used. A team of two students set out in late 2011 to create the tool as their joint MSc-thesis project. We created a meta-model of the

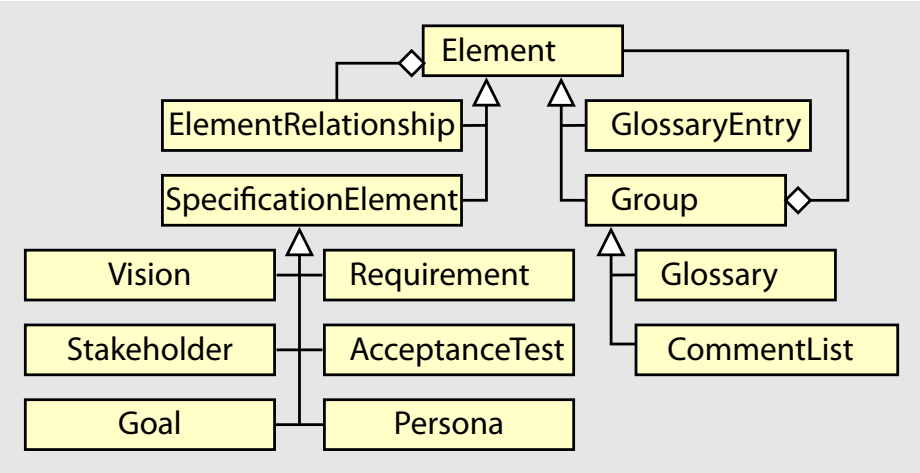

Fig. 1. Excerpt of the meta-model of RED.

concepts in RED (see Fig. 1 for an incomplete overview), and deployed a first version in September 2012.

Reactions to the first version of RED were mixed. While most students acknowledged the need for tool-support, and many saw great potential in RED, substantial shortcomings were also identified. First, RED had been tested mainly on Windows machines, and some major bugs and instabilities on MACs and some Linuxes showed only after a while. Second, RED had no facilities to support group work: the whole project is stored in one big file which conflicted with the distributed and asynchronous working style of any teams in our course. Third, the project was not structured in a way that students could easily contribute to the tool development, fixing bugs or shortcomings on-the-fly. So we have launched a re-engineering effort to address issues one and three, and new thesis projects are launched to address issue two, and completing the feature list to improve the usefulness of RED in the given context.

4 Architecture and Implementation

RED has been created using the latest Eclipse platform available at the time (Eclipse 3.7 "Indigo"). The main rationale behind for using Eclipse is its proven ability to create rich, cross-platform applications. Due to its plugin-architecture, significant leverage through reuse was expected. We also expected beneficial effects towards maintainability and long-term development by adopting a popular framework.

RED has been organized in a set of modules, each providing specific bundles of features: the Core module provides foundational contributions such as the main layout of the application and the meta-model. It supports a number of feature-modules, such as Glossary, SpecificationElements and Help. We have also

reused a number of third-party plug-ins, including EPF RichText and AgileGrid, that increased the code reuse ratio.

5 Features: done, doing, to do

RED offers currently the editors and features shown in the Fig. 2 below. Most requirements have been addressed completely or largely; some shortcomings have been identified with regards to internal software quality (which reduces maintainability), one major missing features (group-work support), and a number of minor issues regarding functionality and usability.

Current work focuses on providing functions for comparing, differencing, and merging RED-files to support group work. We expect this project to complete in the summer of 2013. Besides this, there is ongoing work to improve the on-line help function, provide a manual, and turn the project into a proper open source project to attract more contributions.

Future work will focus on visual editors for (1) goal models, and (2) models of the structure, boundaries, and collaborations of organizations and systems ("context editor"); template-driven editors to allow, e.g., to express requirements compliant with the Common Criteria and some variants of "agile" requirements (i.e., user stories), and an editor for traditional table-structured use cases, including support for project effort estimation with use case points.

Simple Editors	Complex Editors	Features
- Vision	- Personas & Scenarios	- Administrative & tracing information
- Stakeholders	* Cartoon-style	for all elements
- Goals	* Prose-style	- Flexible locking (write/comment/read)
- Glossary	* Structured text	- HTML-Report Generation
- Review Remarks	- Requirements	- Weaving of Model Fragments
- Model Structure	* Prose	- On-line help and serch functions
- Weaving Rel.s	* Test cases	- Full RTF-editor for all input texts
- Scenarios	* Model Fragments	- Enactment of scenarios

Fig. 2. Overview over the features offered in the current version of RED.