

Evaluating the Quality of a UML Business Model

Brian Berenbach

*Siemens Corporate Research, Inc.
Brian.Berenbach@scr.siemens.com*

1. Introduction

Many Siemens operating companies build software products or products with a high software content (e.g. embedded software). Siemens Corporate Research has had the unique opportunity over the last few years to look at the analysis models from which the software products were derived, analyze them, and recommend changes. All the models investigated define commercial products that are or will be in the marketplace in the near future. Because of the size and complexity of the models (some running over a thousand use cases) the internal analysis tool **DesignAdvisor** was created to facilitate their inspection. An analysis was then performed on several models with interesting results.

2. The DesignAdvisor Tool

The DesignAdvisor tool was developed specifically to analyze and measure the “goodness” of large, complex UML models. It has the following features:

- Semantic, user configurable checks of most UML artifacts, including complex, cross-package, cross-diagram rule and metric rules.
- Cross-reference reports, for example, classes and use cases and the diagrams where they appear.
- “Goodness” evaluations for the Analysis, Design, and Implementation models
- A converter that takes an analysis model and creates a traceable “first cut” design model.
- Cross-Model consistency checks
- An advisor to recommend modeling improvements
- An extensible rule set that can be edited and customized
- Export of error reports to XML documents and Excel spreadsheets
- Export/Import of rule set configuration on a per model basis to permit rule tailoring for different kinds of models.
- Seamless operation with Rational Rose™.
- Ability to define 3rd party supplied packages (e.g. J2EE) to be ignored by DesignAdvisor.
- The automatic extraction of a hierarchical set of requirements from the analysis model.

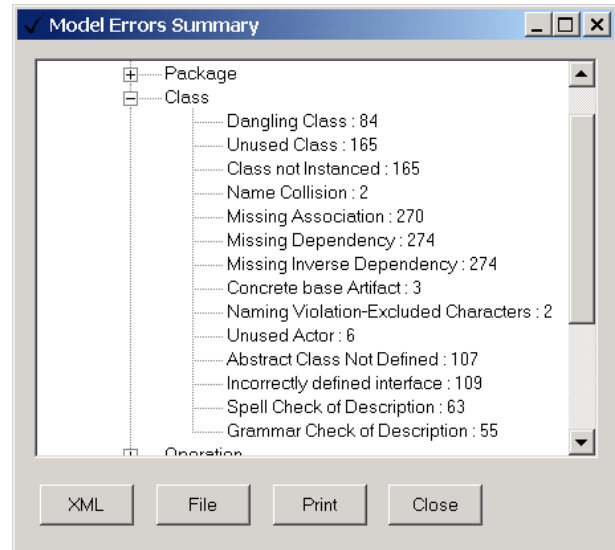


Figure 1 Error Summary Screen

3. Analysis Results

The results of the analysis were most interesting, in some cases even “hair-raising”. For example, it was possible by reviewing the errors in the model to determine:

- The skill level of the modelers,
- the effectiveness of the quality assurance process,
- where there might be design problems or flaws and
- whether the model was sufficiently complete to start design.

Changing names to “protect the innocent”, I will report on my findings in depth, identifying what I think are important lessons learned for any requirements engineering group.

My findings should be of interest to any organization that uses UML models to extract requirements and create software designs.