

A Tool for Enterprise Architecture Analysis of Maintainability

Mathias Ekstedt, Ulrik Franke, Pontus Johnson, Robert Lagerström,
Teodor Sommestad, Johan Ullberg, Markus Buschle

Industrial Information and Control Systems

Royal Institute of Technology

100 44 Stockholm, Sweden

{mek101, ulrikf, pj101, robertl, teodors, johanu, markusb}@ics.kth.se

Abstract

A tool for Enterprise Architecture analysis using a probabilistic mathematical framework is demonstrated. The Model-View-Controller tool architecture is outlined, before the use of the tool is considered. A sample abstract maintainability model is created, showing the dependence of system maintainability on documentation quality, developer expertise, etc. Finally, a concrete model of an ERP system is discussed.

Index Terms: Enterprise Architecture, Probabilistic Relational Models, Software tool, Maintainability

1 Introduction

Today's industrial software systems are increasingly interconnected. A modification of a single system may cause an unexpected cascade effect throughout the rest. Such phenomena pose difficult questions for IT decision makers on how to ensure system maintainability.

In recent years, enterprise architecture has become an established approach for information systems management in organizations. Diagrammatic descriptions of systems and their environment are at the core.

The purpose of creating enterprise architecture models and conducting analyses is to facilitate rational decision making about information systems. For more details on software support to such analysis, see [2].

This paper presents a software tool for analysis of enterprise architecture models. The tool guides the creation of information system scenarios in the form of enterprise architecture models and calculates assessments of the scenarios as they evolve. In the present paper, we focus on a maintainability example.

Enterprise architecture models for evaluating maintainability differ from models used for evaluating system inter-

operability, security, etc. Good maintainability models need to be tailored for that specific purpose.

The first step of enterprise architecture analysis with the tool is to create an *abstract model*, where the theory – concepts and relationships – relevant for the evaluation is specified. This helps users to perform advanced assessments in an automated, easy fashion.

The second step is the collection of data and the subsequent creation of a *concrete model*. The concrete modeler follows the structure prescribed by the abstract modeler, helping the user perform the assessment. The concrete modeler is mainly intended to be used by enterprise architects in the industry. The analyses made in the concrete modeler are based on the mathematical formalism of probabilistic relational models. Probabilistic relational models (PRMs) [1] are a method for probabilistic reasoning over entities and attributes. As it is put in [1], PRMs "are to Bayesian networks as relational logic is to propositional logic".

2 Tool architecture

The tool uses a Model-View-Controller architecture, thus having three types of classes, illustrated in Fig. 1. The data model for abstract and concrete models is specified in XSD files. The data binding tool Castor allows simple marshalling and unmarshalling of XML models to and from java objects (the Model).

The model is contained by the Widget of the Net-Beans Visual Library that also update the view when the model changes. Model operations are controlled by a number of tools. These tools, acting as controllers, provide uniform access to their functionality as they are all implemented in a common way, using the singleton pattern. The use of the singleton pattern enables registration and handling functionality inside the widgets.

The view is implemented as a part of the JApplication framework. The view is in charge of handling all user inter-

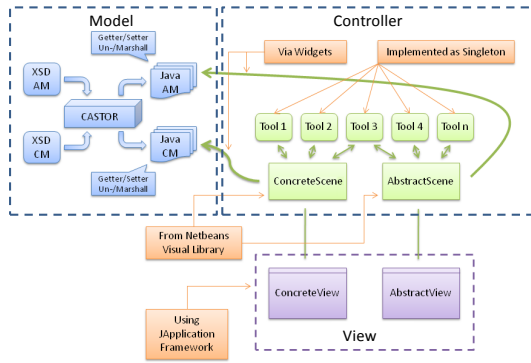


Figure 1. Software architecture sketch.

actions that are not concerned with the drag-and-drop modeling features of the scene. Several methods handle button and menu actions. Also, loading and saving tools are called by the view.

3 Maintainability analysis using the tool

Figure 2 shows the abstract modeler. Here, the theory is encoded in a metamodel. Entities are inter-connected by entity relationships with a specified cardinality. Attributes are interconnected by attribute relationships that correspond to causal dependencies. Uncertainty in these relationships is handled using the PRM formalism. Inter-entity attribute relationships always correspond to entity relationships and are equipped with an aggregation function (Max, AVG, etc.) to cope with cardinality issues.

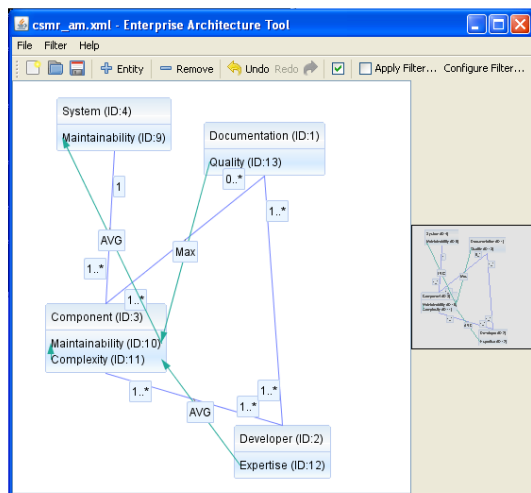


Figure 2. Abstract model for maintainability.

As can be seen in the Fig. 2, a small abstract model example for maintainability analysis could contain the entities System, Component, Documentation, and

Developer. Developer expertise and quality of documentation both causally affect the components complexity and maintainability, which in turn affects the system maintainability. A more thorough description of a "real" abstract model for maintainability analysis can be found in [3].

Figure 3 depicts the concrete modeler where the architectural model is created based on the abstract model. This means that specific instances of the abstract concepts are created to reflect the enterprise at hand. By entering evidence on the states of the attributes and using PRM inference rules, the tool updates the model to reflect the impact of the evidence. The user can thus find updated probability distributions for all attributes in the model.

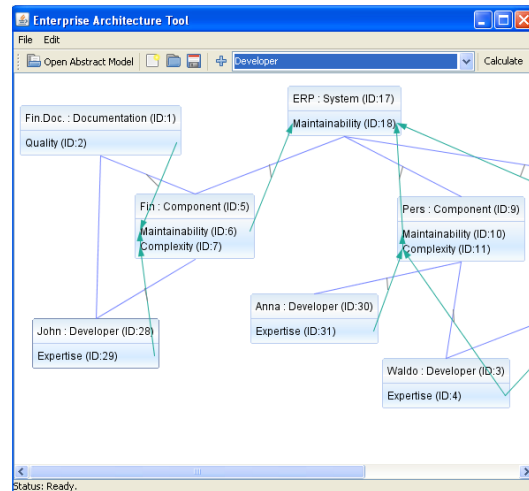


Figure 3. Concrete model for the example.

In this example we have an ERP-system divided into three components, each component being described by one or more documents and developed by one or more persons. Entering information regarding the attributes complexity, quality, and expertise will result in the prediction of a maintainability value for the modeled ERP-system.

References

- [1] L. Getoor, N. Friedman, D. Koller, A. Pfeffer, and B. Taskar. Probabilistic relational models. In L. Getoor and B. Taskar, editors, *An Introduction to Statistical Relational Learning*. MIT Press, 2007.
- [2] P. Johnson, E. Johansson, T. Sommestad, and J. Ullberg. A tool for enterprise architecture analysis. In *EDOC '07: Proceedings of the 11th IEEE International Enterprise Distributed Object Computing Conference*, page 142, Washington, DC, USA, 2007. IEEE Computer Society.
- [3] R. Lagerström and P. Johnson. Using architectural models to predict the maintainability of enterprise systems. In *In Proceedings of the 12th European Conference on Software Maintenance and Reengineering*, Apr. 2008.