ANALYZING SYSTEM MAINTAINABILITY USING ENTERPRISE ARCHITECTURE MODELS

Lagerström, Robert, Royal Institute of Technology, Osquldas väg 12, 100 44 Stockholm, Sweden, robertl@ics.kth.se

Abstract

A fast and continuously changing business environment demands flexible software systems easy to modify and maintain. Due to the extent of interconnection between systems and the internal quality of each system many IT-decision makers find it difficult predicting the effort of making changes to their systems. To aid IT-decision makers in making better decisions regarding what modifications to make to their systems, this paper proposes extended influence diagrams and enterprise architecture models for maintainability analysis. A framework for assessing maintainability using enterprise architecture models is presented and the approach is illustrated by a fictional example decision situation.

Keywords: Enterprise Architecture Analysis, Extended Influence Diagrams, System Maintainability, System Modeling

1 INTRODUCTION

A fast and continuously changing business environment demands flexible software systems easy to modify and maintain. Today many systems at an enterprise are interconnected and a change to one system may cause a ripple effect among the other systems. Numerous systems have also been developed and modified during several years and to make further changes to them demands a lot of effort from the organization, this poses questions such as. Is there enough documentation describing the systems and has the documentation been updated correctly after each change? Is the source code commented and written with good design in mind? These are some of the questions an IT-decision maker faces when wanting to implement changes to systems. It would be useful for IT-decision makers to be able to analyze how much effort a certain change to a system would require, but in a large enterprise it can be difficult to assess this effort since there are often many systems, persons, and processes involved in a change.

During the last decade, enterprise architecture has grown into an established approach for software system management. Enterprise architecture is model based in the sense that diagrammatic descriptions of the systems and their environment constitute the core of the approach. Some enterprise architecture models contain information about the maintainability of a system, thus one could use enterprise architecture models for system maintainability analysis.

This paper proposes an approach for system maintainability analysis using enterprise architecture models and extended influence diagrams. The maintainability framework presented in this paper, in the form of an extended influence diagram, consists of attributes affecting the maintainability of software systems. The values of the attributes in the framework can be gathered from enterprise architecture models and with the mathematics in extended influence diagrams a quantitative value of the maintainability effort can be calculated and used in decision situations regarding what modifications to make to a system. Figure 1 presents this approach graphically; the analysis part is depicted as a machine, containing an extended influence diagram, receiving enterprise architecture models as input and presenting the analysis result as output.
Figure 1. The approach of using enterprise architecture models in order to analyze system qualities such as maintainability.

This paper unfolds as follows. Section 2 contains information about how to perform enterprise architecture analysis using extended influence diagrams. Then the framework for system maintainability is presented in Section 3, the framework consists of an extended influence diagram and a meta-model. In Section 4 an example analysis is demonstrated using the proposed maintainability framework. Finally, the paper is concluded in Section 5.

2 ENTERPRISE ARCHITECTURE ANALYSIS

The purpose of having enterprise architecture models and conducting analysis of these is to facilitate rational decision making regarding software systems at an enterprise. In the paper *Extended Influence Diagrams for Enterprise Architecture Analysis* by Johnson et al. (2006) a language called extended influence diagrams was introduced as a way of modeling goals, like maintainability, for enterprise architecture analysis.

Extended influence diagrams graphically represent a network of nodes and relations. The different relations that can be used in an extended influence diagram are either causal, informational, or definitional relations. The nodes represent variables, and there are three different types of nodes: decision nodes, chance nodes, and utility nodes. Decision nodes represent the decisions that can be made, e.g. selecting between different system scenarios. Utility nodes represent the goals, e.g. system maintainability. Chance nodes represent all variables related to the utility node, e.g. system size and system complexity. The syntax for the different relations and nodes is presented in Figure 2. Extended influence diagrams are an extension of Influence diagrams, thus inheriting the Bayesian mathematics. More information on Influence diagrams can be found in (Howard 1983) or (Shachter 1988). The nodes in extended influence diagrams have conditional probability matrices related to them, these matrices represent the states each node can have and the probability of each state occurring. An example extended influence diagram with a conditional probability matrix is presented in Figure next to the extended influence diagram syntax. In this example the probability matrix represents the probabilities of the attribute System Size to be small, medium, or large if a Scenario X or a Scenario Y is selected. As can be seen in the figure the System Size is medium with a probability 0.8 (80 %) if Scenario X is selected.
In the approach of using extended influence diagrams for enterprise architecture analysis, an extended influence diagram is related to enterprise architecture models through the entities, sometimes referred to as classes, and the attributes in the models. This relation between an extended influence diagram and an enterprise architecture model is visualized in Figure 3. What entities and attributes a model should contain to enable different analyses is determined by the extended influence diagram one chooses to use for a certain analysis. The utility node in the extended influence diagram represents the goal of the analysis, e.g. information security, performance, or maintainability. When the analysis goal is set, i.e. the extended influence diagram is selected, each node in the extended influence diagram relates to one attribute in the model, see Figure 3.

There are several meta-models, often called viewpoints, used in enterprise architecture modeling. For example, one could use the meta-models presented by Cummins (2002), O’Rourke (2003), Lankhorst (2005), and Niemann (2006). Common for all meta-models is that they contain entities, but what is often missing are the attributes of these entities. Lankhorst (2005) presents a viewpoint called infrastructure usage viewpoint which contains the entities hardware, supporting system software, networks, applications, and services. There are no attributes for these entities presented, e.g. the network entity could have the attribute performance or the supporting system software entity could have the attribute number of lines of source code. It is proposed by Johnson et al. (2006) that the enterprise architecture meta-models should contain both entities and attributes, and that the attribute values should be presented in the form of conditional probability matrices like the nodes in the extended influence diagrams, thus forming a relation between the extended influence diagram and the model. The relation between models and extended influence diagrams is visualized in Figure 4. In the figure an enterprise architecture model of a system Scenario X has an entity called System A. This entity in turn has the attribute Size. The values of the attribute Size of System A is presented in the form of a conditional probability matrix, this matrix forms the connection to the node called System Size in the extended influence diagram.
3 MAINTAINABILITY FRAMEWORK

In this section we apply the approach of analyzing different system qualities with extended influence diagrams and enterprise architecture models on the system quality maintainability. Section 3.1 presents the extended influence diagram for maintainability and 3.2 presents the meta-model used when modeling for maintainability analysis.

3.1 Extended influence diagram for system maintainability

System maintainability is defined as the ease with which a software system or component can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment (IEEE 1990). To simplify assessments of maintainability researchers and practitioners have been looking for the attributes affecting maintainability for a long time and several frameworks have been proposed, such as Oman (1992), Chan (1996), Granja-Alvarez (1997), ISO (2001), Aggarwal (2002), and Matinlassi (2003). The maintainability framework presented in this paper is to a large extent influenced by the work of Oman et al. (1992).

System maintainability is said to be affected by; the maturity of the system’s development and maintenance personnel, the maturity of the development and maintenance processes, the quality of the system’s supporting documentation, the system’s architectural quality, the platform’s quality on which the system executes, and the system’s source code quality. Since these variables are difficult to measure, they have been further broken down into more easily measurable attributes. For example the maturity of the system’s development and maintenance personnel is measured in the staff’s level of experience with development and maintenance work, the staff’s level of language expertise on the programming languages used within the system, and finally the staff’s level of knowledge on the system they are maintaining. In Figure 4 the extended influence diagram for system maintainability is presented, further information on the framework can be found in (Johnson 2007).

![Figure 4. The extended influence diagram for maintainability.](image-url)
3.2 Meta-model for system maintainability analysis

The meta-model used when modeling for system maintainability analysis should be able to represent the information in the maintainability extended influence diagram, i.e. the values of all nodes in the extended influence diagram affected by the decision node should be collected from the models that are based on the maintainability meta-model. The maintainability extended influence diagram presented in Figure 5 has nodes related to persons, processes, documents, systems, platforms, and source code. Thus these are the entities needed in the meta-model. Looking at different meta-models proposed by other authors it is clear that the entities needed in a maintainability meta-model could be found in several already existing meta-models. On the contrary the attributes for each entity matching the nodes in the extended influence diagram are more difficult to find in others work, e.g. that the entity Source Code has the attribute Level of Coupling. Because of this a new meta-model was developed, this meta-model is influenced by already existing meta-models but with the addition of attributes for the entities. Figure 5 depicts the meta-model, which has six entities and each entity has several attributes all related to nodes in the extended influence diagram for maintainability.

4 MAINTAINABILITY EXAMPLE ANALYSIS

This section presents an example analysis employing the approach of enterprise architecture analysis using the maintainability extended influence diagram and meta-model. Firstly, the fictional enterprise and its IT-decision maker are introduced, and then the decision situation and its future system change scenarios are modeled. Finally, the analysis and results of the example are presented.

4.1 Enterprise and IT-decision maker background

As a result of almost a decade of mergers and acquisitions on the deregulated European electricity market, Southern Energy is today one of Europe’s largest energy companies, acting in all parts of the electricity value chain. The company has a large amount of IT systems required to ensure a non interrupted flow of electricity within the Southern Energy’s distribution network. There is a dedicated CIO making decisions related to IT, the CIO is constantly involved in, and supervises, the evolution of the enterprise’s information systems. He has two enterprise architects working within the company under his command. The enterprise architects spend a considerable amount of time each week on
continuous updating of the models that describe the enterprise architecture and also assisting the CIO in analysis of the models’ content.

Southern Energy’s geospatial network system, GeoNet, needs some modifications due to changed requirements. The CIO has two changes he wants to implement, but his budget only allows him to do one. The first change scenario that the CIO would like to implement is to move the facilities record application in GeoNet to an independent database. The second change is moving the grid calculations application to a custom made calculation system. The CIO would like to analyze the effort of making these two changes, and compare the effort between the two scenarios so he could choose the scenario requiring the least effort. The CIO and his two enterprise architects chooses to use the maintainability extended influence diagram and meta-model proposed in this paper and described in Section 3. Based on the maintainability meta-model, the CIO selects the enterprise architecture models that can be employed in order to assess the maintainability. The enterprise architects needed to add some information in the models and thus had to collect some data regarding the system GeoNet and its environment. They found some documentation regarding the system and also performed some interviews with maintenance and development personnel. The enterprise architecture model of the first change scenario for GeoNet is presented in the following subsection.

4.2 Change scenario 1: Moving the facilities record application from GeoNet

In the first scenario, moving the facilities record application in GeoNet to an independent database, the enterprise architects collected information related to this change based on the attributes in the meta-model. The enterprise architects found that the GeoNet system executes on a Linux platform and that the facilities record application source code is compiled into the application affected by the change. Further they found that there is a process for maintaining the facilities application and that Mr. Andersson is the person making changes to this application. There is also a design specification related to the facilities record application. Further the architects found that GeoNet exchanges data with several other systems; a network calculation system, an ERP system, a CRM system, and a SCADA system.

Data regarding the attributes in the meta-model was gathered by the enterprise architects and inserted in the model. The data was accordingly also inserted into the extended influence diagram to calculate the maintainability, i.e. the effort of implementing the change. For example the attribute Platform Quality was calculated from the attributes Availability, Porting Tools, and Level of Standardization. The Availability attribute was measured by the enterprise architects to be Low with the probability 0.1, Medium with the probability 0.8, and High with the probability 0.1. This means that the availability of the platform is Low with 10 % certainty, Medium with 80 % certainty, and High with 10 % certainty. Based on the three measured and modeled attributes of the platform, its quality was calculated in the extended influence diagram with the inherent Bayesian mathematics. The quality of GeoNet’s Linux platform was calculated to be Low with the probability 0, Medium with the probability 0.84, and High with the probability 0.16.
In the second scenario, moving the grid calculations application to a custom made calculation system, the CIO’s enterprise architects collected similar data as in the first scenario and created a model using the meta-model for maintainability. The similarities between the two scenarios are the attributes of the system GeoNet’s architecture and the Linux platform it executes on. In the second scenario there is another maintenance process related to the calculation application, and there are two resources related to this process, Mrs. Williams and Mr. Smith. There is only one document relevant for the assessment of the second scenario, the calculation application design specification. Further there are two different source code files, one for the calculation application and one for the calculation presentation application. The collected information about the second scenario and the calculated values of the attributes were presented in an enterprise architecture model like the one for the first scenario.

4.4 Maintainability analysis of the two change scenarios

Using software for Bayesian analysis such as Genie (2007) or Hugin (2007), the maintainability of the two scenarios described in the above subsections was calculated and compared. The maintainability of Scenario 1 was evaluated to be Low with the probability 0, Medium with the probability 0.107, and High with the probability 0.893. The maintainability of the second scenario was calculated to be Low with the probability 0, Medium with the probability 0.793, and High with the probability 0.207. The results are presented in Figure 7. The first diagram in the figure visualizes the expected mean values of the scenarios; where Scenario 1 is calculated to 95 % and Scenario 2 to 60 %. Since the collected data is not 100 % certain, there is also an uncertainty related to the scenario values, this is visualized as thin bars in the first of the three diagrams. For more information on the calculations the interested reader is referred to (Howard 1983), (Johnson 2007), and (Shachter 1988).

The results from the two change scenario evaluations can now be compared to aid the CIO of Southern Energy in selecting the change requiring the least effort. In Figure 7 three diagrams visualizes the results of the maintainability analysis; the first diagram presents the comparison of the two scenarios including the uncertainty of the assessment, the second diagram presents the probabilities of Scenario 1’s values, and the third diagram presents the probabilities of Scenario 2’s values.
The results presented provide support for the CIO in the choice between the two scenarios. As can be seen in the diagrams the CIO should make the choice of implementing change scenario one, to move the facilities record application in GeoNet to an independent database. In the first of the three diagrams there is a level of uncertainty associated with both scenarios’ maintainability. The uncertainty of the results could be decreased by engaging in more data collection, i.e. by increasing the precision of the enterprise architecture models. This would provide better decision support for the CIO but on the other hand require more resources.

5 CONCLUSIONS

This paper proposes the approach of employing extended influence diagrams and enterprise architecture models when analyzing system maintainability. A framework for maintainability was provided, in the form of an extended influence diagram and a meta-model to be used when modeling systems for maintainability analysis. The extended influence diagram and the meta-model were then used in a fictional example to illustrate how these can aid in decision situations.

References


