Integration of Decision Analysis in Process Life-Cycle Models

Vishesh Duggar
College of Computer and Information Science
Northeastern University
Boston, Massachusetts
vishesh.duggar@gmail.com

Kenneth Baclawski
College of Computer and Information Science
Northeastern University
Boston, Massachusetts
kenb@ccs.neu.edu

ABSTRACT

Decision analysis is an important part of development processes, and decision analysis techniques have been very highly developed. However, the connection between a particular decision analysis and the process within which it occurs is informal at best. Consequently, it can be difficult to revisit or reuse a decision analysis at a later time or for another purpose. Once a decision has been made, the analysis that was performed is often discarded, and even if it is retained, the context within which the analysis was performed is not documented sufficiently for the analysis to be easily revisited when the situation changes or to be reused for similar decisions. In this position paper we propose formal annotation of decision analyses using Semantic Web technologies, and the development of a formal process model using the Eclipse Process Framework (EPF) to document the roles, artifacts and processes that are used by decision analyses. Formal annotation also makes it possible to reason about the analysis and process, allowing one to check consistency and to answer queries. The use of EPF makes it possible to integrate the decision analysis process as a component in the process where it is used, and to reuse the decision analysis in other contexts.

Categories and Subject Descriptors

D.2.12 [**Software Engineering**]: Interoperability; D.2.2 [**Software Engineering**]: Design Tools and Techniques; G.3 [**Probability and Statistics**]: Stochastic Processes

Keywords

decision analysis, influence diagrams, Bayesian networks, decision support, Semantic Web

1. INTRODUCTION

In any development process there are numerous instances where one must make a decision. This is especially true for software engineering processes, where decisions occur at virtually every level of granularity. At the higher levels of granularity and management, the decision process is especially important because of the impact that the decision can have on the success of the project. For this reason, such decisions should make use of decision support tools, especially tools that incorporate risk assessment[8]. However, even when an elaborate decision support tool is being used, the connection between the decision process and the development process can be informal.

A further problem with decision support as it is often practiced is the lack of reusability. For certain classes of decision, where tools have been developed, such as resource decisions [3], one can make use of sophisticated models that incorporate Bayesian networks, empirical data and expert judgement. Such models and the tools for them are very powerful and effective. However, if a particular decision does not fit exactly into the framework of one of these classes, then a more generic approach must be taken, and it can be difficult to reuse such decision analyses when a similar decision must be made.

In this paper, we propose to use formal representations of decision analyses to deal with these issues. Our approach is to use Semantic Web techniques for the formal representation of the decision analysis and the Eclipse Process Framework (EPF) for the decision process. In this way, a formal artifact can be produced that expresses how and why a decision was made, which can be linked into the many other artifacts of the development process. Semantic Web search tools may then be employed to find similar decisions so that they may be reused.

An important advantage of this approach is the ability to reason about the decision analysis within the context of the development process. The decision analysis itself represents a sophisticated reasoning process, but it can be invalidated if circumstances governing the decision evolve over time. According to Kevin et al.[12], "...part of the value of typical software product, process or project is in the form of embedded options. These real options provide design decision-makers with valuable flexibility to change products and plans as uncertainties are resolved over time." If the decision analysis was not formally represented and linked with the development process, it may not be realized that it is necessary to reconsider the decision,

Possibly the most dramatic example of this was a design decision for the Ariane 4 rocket software that was not re-

International Workshop on Living with Uncertainties November 5, 2007, Atlanta, Georgia, USA

considered for the Ariane 5 rocket. The result was that the rocket crashed on its first launch[4].

This paper begins with some brief background introducing the technologies we will be employing, including the Semantic Web, EPF and influence diagrams. We then describe the methods we will use. The first step is to develop an ontology using Protégé[10]. Once the ontology is in place it is used as the basis for the development of process descriptions. Our emphasis is on the integration of decision analysis into the software development life-cycle, not on any particular decision analysis technique or life-cycle model. The hope is that the ontologies and process descriptions may be used for any decision analysis techniques and any life-cycle model.

2. BACKGROUND

In this section we give some background about Semantic Web technology, EPF and influence diagrams.

2.1 Semantic Web

One of our goals is to document decision analysis so that it can be easily reused both by humans and software agents. This is done with a formal representation of terminology called an ontology. An ontology defines the terms used to describe and represent an area of knowledge. Ontologies are used by people, databases, and applications that need to share domain information (a domain is just a specific subject area or area of knowledge, like medicine, tool manufacturing, real estate, automobile repair, financial management, etc.). Ontologies include computer-usable definitions of basic concepts in the domain and the relationships among them They encode knowledge in a domain and also knowledge that spans domains. In this way, they make that knowledge reusable. Semantic Web ontology tools can perform automated reasoning using the ontologies, and thus provide advanced services to intelligent applications such as: conceptual/semantic search and retrieval, software agents, decision support, speech and natural language understanding, knowledge management, intelligent databases, and electronic commerce. [6, 2]

2.2 Eclipse Process Framework

The process model we use is expressed using the Eclipse Process Framework Composer. EPF is a tool for producing a customizable software process engineering framework, with exemplary process content and tools, supporting a broad variety of project types and development styles. [5]. The extensible process frameworks bundled with EPF Composer are OpenUP/Basic, OpenUP/MDD, and recently added "other agile components." We are not going to extend any of those but create a new process model using EPF Composer.

EPF was derived from RUP or Rational Unified Process. A subset of RUP that was released by IBM to the open source community is known as OpenUP. OpenUP is a minimally sufficient software development process - meaning that only fundamental content is included. Thus, it does not provide guidance on many topics that projects may deal with, such as large team sizes, compliance, contractual situations, safety or mission critical applications, technology-specific guidance, etc.[5] RUP/OpenUP is an iterative software development process framework. It is not a step by step process cycle but an adaptable process framework which allows an organization or a development team to put together a process model according to their needs. Such a framework was and is needed desperately as no single software development life cycle is suitable for all the software development.

What EPF or RUP essentially do is, provide us with a tool through which we can define the method content separately. Method Content as the name suggests is the description of goals, required skills, various artifacts and deliverables etc. Without worrying about where and when they are used in the process life cycle. After defining the method content the processes are defined using the method content. The processes describe the sequence in which the work is performed by different roles and the deliverables that would be the delivered during the course of the process life cycle. The important thing to note here is that the work, roles and deliverables are all defined in the method content.

The method content and processes can be re-used as it is. They can also be configured according to the need of a project and then re-used. They can be extended if there seems to be a need for it. Or they can be modified a little to meet the requirements. Separation of the definition of the method content and processes enables one to re-use, extend or tailor them independently.

2.3 Influence Diagrams

Uncertainty is an inevitable part of any decision making process. Influence diagrams are a systematic, empirical technique for representing uncertainty in a decision. The formal basis for influence diagrams is probability theory, and they are represented using Bayesian networks. A Bayesian network (BN) is a graphical mechanism for specifying the joint probability distribution of a set of random variables[9]. As such, BNs are a fundamental probabilistic representation mechanism for stochastic models. The use of graphs provides an intuitive and visually appealing interface whereby humans can express complex stochastic models. This graphical structure has other consequences. It is the basis for an interchange format for stochastic models, and it can be used in the design of efficient algorithms for data mining, learning, and inference. The range of potential applicability of BNs is large, and their popularity has been growing rapidly. BNs have been especially popular in biomedical applications where they have been used for diagnosing diseases[1], among many other applications.

The random variables of the Bayesian network are represented as nodes of a graph. The edges denote dependencies between the random variables. This is done by specifying a conditional probability distribution for each node. It is also required that the edges of a BN never form a directed cycle: a BN is acyclic. If two nodes are not linked by an edge, then they are conditionally independent. One can view this independence property as defined by (or a consequence of) the following property of a BN: The JPD of the nodes of a BN is the product of the CPDs of the nodes of the BN. This property is also known as the chain rule of probability. This is the reason why the BN was assumed to be acyclic: the chain rule of probability cannot be applied when there is a cycle. When the BN is acyclic one can order the CPDs in such a way that the definitions of conditional probability and statistical independence can be applied to get a series of cancellations, such that only the JPD remains.

"An influence diagram is a way of describing the dependencies among aleatory variables and decisions. It can be used to visualize the probabilistic dependencies in a decision analysis and to specify the states of information for which independencies can be assumed to exist." [7] It is derived from Bayesian Networks, but has a very high degree of annotation. This is a key requirement for understanding and re-using a decision analysis.

An influence diagram is a network comprising of directed graph which has three types of nodes. It has at most one value utility or value node that represents the quantity that is to be maximized. It also has zero or more chance nodes representing random variables. And lastly, it may have zero or more decision nodes, which represent the alternatives available to the decision maker. [11]

The diagram as a whole encodes the criteria. Evaluating the diagram for each alternative to get the utilities for comparison to reach a decision would be the argument. There are various algorithms that have been proposed for evaluating influence diagrams. Schachter developed an algorithm that can evaluate any well formed influence diagram and determine the optimal policy for its decision.

3. METHOD

We now discuss the methods we will be using. Starting with ontology development as the basis, we then develop decision analysis process models.

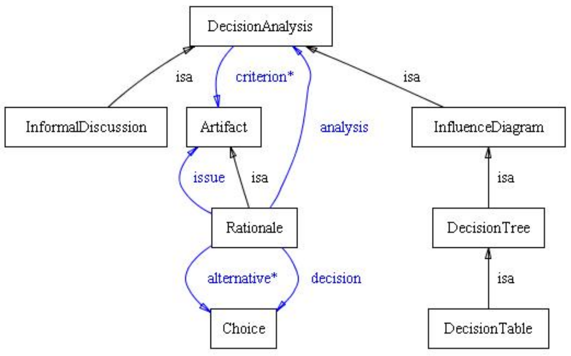
3.1 Ontologies

The description of a decision analysis, including the reasons underlying a decision, will be called a *rationale*. The main constituents of a rationale are:

1. Issue
2. Analysis
3. Criteria
4. Alternatives
5. Decision/Suggestion

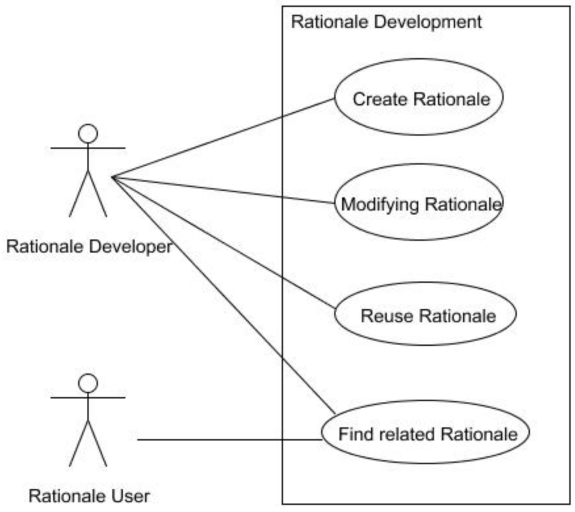
The issue is the matter at hand that has to be decided. The analysis includes all relevant background information and their relationships to one another. When stated informally, the analysis consists of the statements made based on one of or all criteria, to support or oppose one or more alternatives. When stated formally, the analysis consists of a diagram or table that expresses to what extent the criteria support the alternatives. Criteria are the requirements that are necessary for selecting a particular choice among all the alternatives. Alternatives are the choices that could be made to make a decision. A formal decision analysis may be expressed using influence diagrams or other mechanisms such as decision trees or tables. Decision/Suggestion is a description of the final decision or suggestion made. These constituents are used later to define the process model.

The tool we are using for ontology development is Protégé[10]. Our initial ontology, at the class level, has the following diagram:



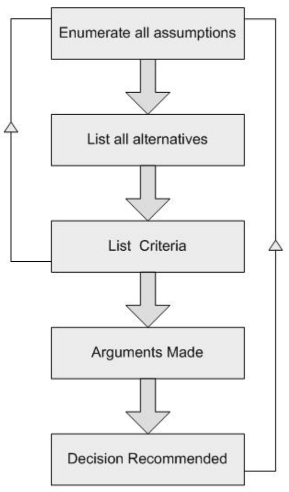
3.2 Process Model

The process model in essence implements a basic decision making sequence of processes but it extends it by specifying a model to document it. The use case diagram showing the actors and activities during formal documentation of decision analysis is:



The two roles/actors are the developer of the decision analysis documentation and the user of the decision analysis. The user is the actual decision maker who works with the developer to ensure that the decision analysis is properly integrated with the development process. The developer can perform a number of actions on the repository of rationales, such as create, modify and reuse/repurpose. In collaboration with the rationale user, the developer helps to find relevant rationales. There are many other use cases that we have not developed that are concerned with activities such as reconsidering decisions and inference.

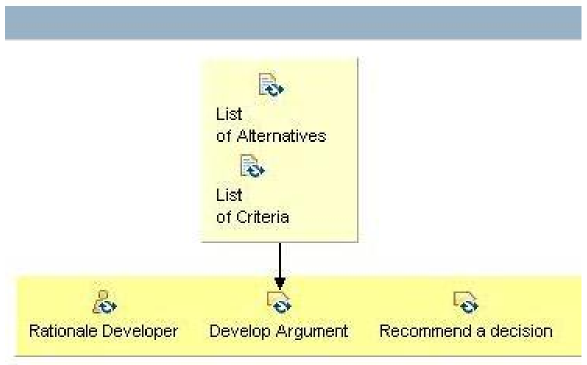
The process model is:



The process involves a number of steps and iterations as follows:

1. Enumerate all the assumptions that are relevant and can be inferred based on the context or situation.
2. Exhaustive list of all the alternatives that can be chosen for a particular decision have to be documented.
3. Similarly, a list of criteria based on which any alternative would be chosen for an issue is documented.
4. Both step 1 and step 2 are done iteratively till a satisfied list of both alternatives and criteria are available.
5. Relevant arguments for each alternative based on the list of criteria are obtained.
6. Based on the arguments put forward a decision is recommended.
7. The whole process from steps 1 to 6 could then be iterated till a satisfactory decision is obtained.

The following shows one of the EPF diagrams:



In this diagram the Rationale Developer is a role that performs two activities: Develop Argument and Recommend a Decision. The inputs to the activities are two kinds of artifact: List of Alternatives and List of Criteria.

Acknowledgements
Kenneth Baclawski was partially supported by Vistology, Inc., Framingham, Massachusetts.

4. REFERENCES

[1] M. and Jordan. Variational probabilistic inference and the QMR-DT network. *J. of Artif. Intell. Res.*, 10:291–322, 1999.

[2] K. Baclawski and T. Niu. *Ontologies for Bioinformatics*. MIT Press, Cambridge, MA, October 2005. ISBN 0-262-02591-4.

[3] N. Fenton. Making resource decisions for software projects. *Proc. 26th Intern. Conf. Soft. Engineering*, pages 397–406, 2004.

[4] J. Gleick. A bug and a crash: Sometimes a bug is more than a nuisance. *New York Times Magazine*, December 1 1996.

[5] P. Haumer. Eclipse process framework composer : Part 1 key concepts, April 2007. http://www.eclipse.org/epf/general/EPFComposerOverviewPart1.pdf.

[6] J. Heflin. Owl web ontology language: Use cases and requirements. http://www.w3.org/TR/webont-req/.

[7] R. Howard and J. Matheson. Influence diagrams. *Decision Analysis*, pages 127–143, September 2005.

[8] W. Jiamthubthugsin and D. Sutivong. Resource decisions in software development using risk assessment model. *Proc. 39th Hawaii Intern. Conf. on System Sciences*, pages 229–236, 2006.

[9] J. Pearl. Graphical models for probabilistic and causal reasoning. *Handbook of Defeasible Reasoning and Uncertainty Management Systems, Volu me 1: Quantified Representation of Uncertainty and Imprecision*, pages 367–389, 1998.

[10] Protege. Protégé website, 2004. `protege.stanford.edu`.

[11] R. Shachter. Evaluating influence diagrams. *Operations Research*, 34(6):871–882, Nov–Dec 1986.

[12] K. Sullivan. Software design as an investment activity: A real options perspective. *Real Options and Business Strategy: Applications to Decision Making*, pages 215–261, 1999.