XML-based SCM Environment and Related Metrics

Ya-Chi Weng and Chin-Feng Fan
Computer Science and Engineering Dept., Yuan-Ze U.
csfanc@saturn.yzu.edu.tw

Abstract- *This paper presents an XML-based SCM (Software Configuration Management) environment along with XML-based data collection for metrics. We propose using XML tags and links to express semantic relationships among configuration items and their contents so as to efficiently assist major SCM activities, such as impact analysis and configuration verification. Furthermore, information retrieved from XML links/tags can be modeled using BBNs (Bayesian Belief Networks) for metrics measurement. The proposed approach can effectively support SCM activities and its related decision-making.*

Keywords: XML, Software configuration management, BBN (Bayesian Belief Networks)

1. Introduction

Current Software Configuration Management (SCM) tools usually manage various types of files, and support changes and different versions of source code. But these tools normally do not address the relationships between the contents of different configuration items. However, such inter- or intra-relationships among configuration items can be used to support SCM functions more effectively than approaches without using them. XML can easily implement relationships by marking and linking related portions with meaningful user defined names. Thus, we propose to use XML to annotate important relationships among contents of configuration items to effectively support SCM major activities. Moreover, we propose to combine the information extracted from XML tags and links with Bayesian Belief Networks (BBN) to estimate software maintainability, complexity, and to support release decision.

In the following, we will first give a brief background introduction, followed by our XML approach and tools description. Then XML-based metrics are presented. Finally, a conclusion is given.

2. Related Background

2.1. SCM tools

Many commercial SCM tools are available. For instance, IBM Rational ClearCase [3] helps users manage and track software resources; Visible system [8] produced by Razor supports problem tracking, file version control, and the release management; while PVCS (Project Version Control System) [6] assists Version Manager and Tracker.

Most of the current SCM tools focus on management of different versions of source code; our SCM environment supports complete SCM activities. Most of the current SCM tools do not explicitly handle relationships in the contents of configuration items; yet, our tools deal with such inter- or intra- relationships so as to accurately and efficiently support SCM functions.

2.2. Bayesian Belief Network

Bayesian Belief Network [5] is an acyclic graph used for modeling and reasoning with uncertainties. Each node in a BBN represents a random variable, whose state is usually expressed in discrete numbers or ranges. Each edge in the graph represents the casual influence between connected nodes. A Conditional Probability Table (CPT) is associated with each node to denote such casual influence. CPT's are filled by experts or inferred from statistical data. Once new evidence is obtained, it can be plugged in the graph to update the states of the related nodes. The calculation is propagated from parent nodes to child nodes and vice versa. A BBN graph can be expanded into an influence diagram by adding decision nodes and utility nodes. The former are shown by rectangles; the latter, representing cost or profit functions are depicted by diamonds. Figure 1 is a sample BBN example. In short, BBNs provide powerful modeling and computation under uncertainties. Thus, we use BBN's such advantages to estimate SCM related metrics.

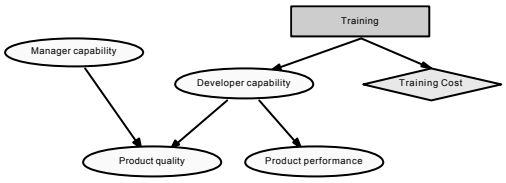

Figure 1. BBN example

3. Our XML-based SCM Environment

In the following, we will first introduce our XML-based SCM environment, and then present its related metrics, a by-product extracted from XML tags.

We have constructed an XML-based SCM environment on Windows 2000, using Visual Basic 6.0 and Microsoft Access.

Our environment provides different tools to assist the following SCM tasks [4]:

1) Configuration ID and Markup
   - DTD definition tool
   - Markup tagging tool
2) Configuration Control
   - Workflow tool
   - Change impact analysis tool
3) Configuration Status Accounting
   - Status accounting reports tool
4) Configuration Verification
   - Review tool
5) Release Management (provides related metrics)

Our tool allows the user to define his/her own tags and links by defining DTD's (Document Type Definitions), and then mark up related documents using the selected DTD. The reviewer may use XML DTD to define requirements of related industrial standards. Thus, documents marked by such tags can be checked for its conformance to the standards. Figure 2 is a sample DTD diagram for requirements specification.

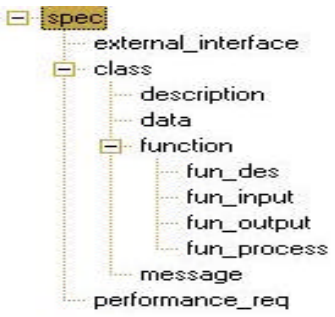

Figure 2. DTD diagram for requirements

In addition, to support the implementation of a full-scale software configuration environment, our system also provides the following predefined XML tags and links to express relationships among configuration items. We identify the related relations including *tracing*, *reference*, *use* and *inheritance*. They are explained below:

1. *Tracing* – It indicates tracing relations at contiguous stages. This is shown in Figure 3. Tracing links may assist reviewers to perform configuration verification.
2. *Reference* – Documents may refer to figures or document fragments from different sources.
3. *Use* – It presents as caller and callee relations.
4. *Inheritance* – It presents superclass and subclass relations for an object-oriented system.

UML diagrams can also be tagged by their types, messages, and use cases, etc. Besides, we use XML tags to show change history. The changed fragments may be marked using new/removed/changed tags. The change related DTD is shown in Figure 4. The CR No. (Change Request Number) identifies the change request form.

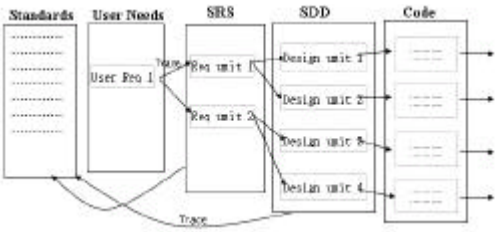

Figure 3. Tracing relationship between documents

```
<! ELEMENT CR No. (#PCDATA)>
<! ATTLIST CR No. type (New|Removed|Changed)
#REQUIRED>

<!ELEMENT simplelink ANY>
<!ATTLIST simplelink
    XLink:type (simple) #FIXED "simple"
        XLink:href CDATA #IMPLIED>
    Xlink:title CDATA #IMPLIED>
```

Figure 4. The change related DTD

Configuration identification is the first SCM activity. It assigns unique identifiers to software configurations items. In order to help users manage concerned files, we add SCM-related identifiers as tags, such as names of the file, versions, ID's, and information of related files, as the header of a document.

Configuration control is a major SCM activity. The validity and impact of a suggested change are analyzed before the change request is forwarded to the CCB (Change Control Board). Once the suggested change is approved, it will be sent to the developed team for change implementation and finally the reversion has to

be verified. Our system provides a workflow solution to control change process via e-mail. Change request forms will automatically deliver via e-mail to the designated receivers according to the customized workflow. The change request forms should include a unique number and be filled in by involved persons in order. The change request form forwarded by workflow is shown in Figure 5.

Our tools retrieve related information using related links as *use* and *inheritance* to assist analysts to perform impact analysis. *Use* links link caller-callee relations, and *inheritance* links indicate superclass-subclass relations. We utilized "use" and "inheritance" relations to support analyzers in change impact analysis.

Configuration status accounting records and reports information related to a software configuration item. Users can retrieve the information of a SCM item from the marked header.

Configuration verification is another major SCM activity, which verifies that the software system matches the description in the specifications and related documents. The results of verification are also kept by our tool since they may be used for release decision. Furthermore, our tools can retrieve related portions using tracing links or change/modification links to assist the reviewer to perform configuration verification. The tracing links can be used to extract related contents of documents at contiguous stages to assist reviewers. The tracing mode is shown in Figure 6.

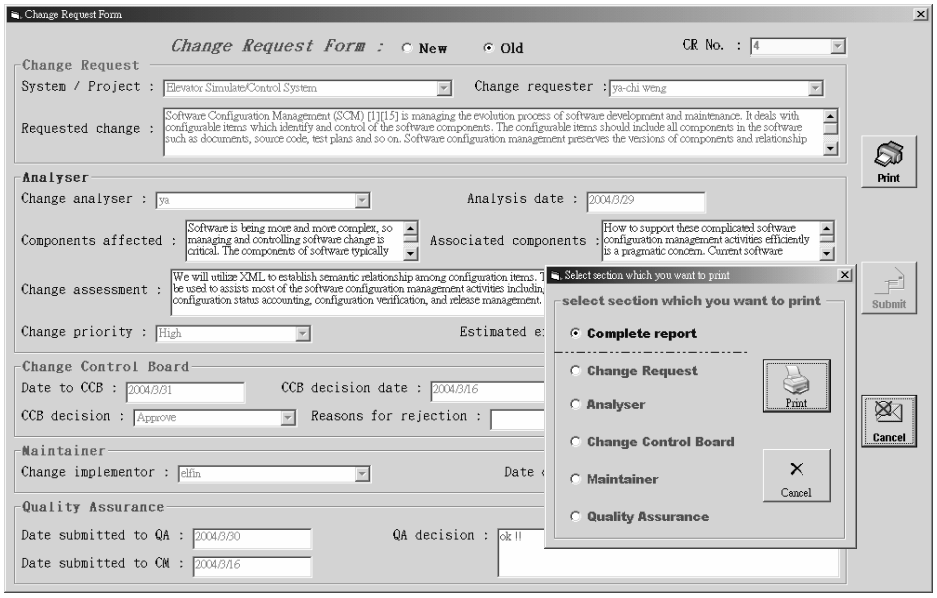

Figure 5. Change request forms in change control workflow

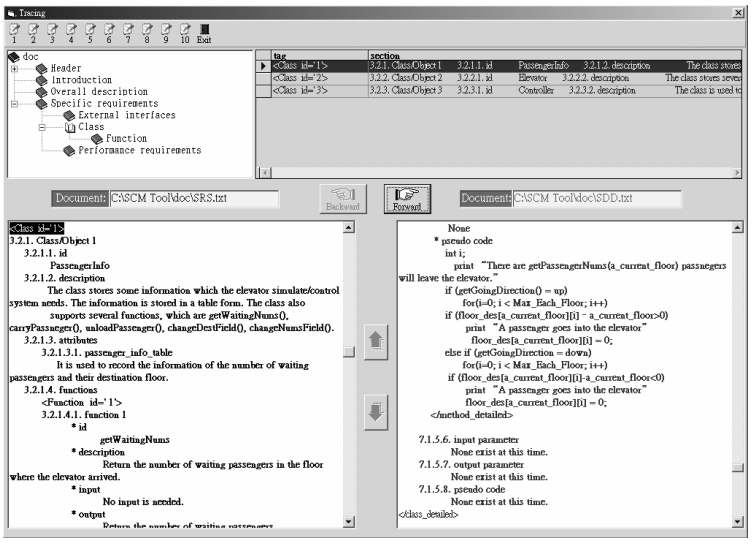

Figure 6. Configuration verification (tracing between requirements and design)

4. XML-based Metrics

Once a project has used our SCM environment to maintain its configuration items, the information extracted from XML tags and links can be collected for metrics measurement. We use the following ways to estimate software complexity, maintainability, and to make release decision:

(1) Weighted summation: When related factors all have numerical values, we simply use weighted summation for measurement.

(2) Use BBNs : When most of the concerned factors are uncertain, BBNs can be used for metrics estimation.

(3) Hybrid approach: When the number of certain factors are more than that of uncertain ones, the uncertain ones are first estimated by BBNs and then BBN results are combined with the certain factors using a weighted summation.

These methods are described in the following subsections.

4.1. Complexity measurement

A wide variety of complexity-related metrics has been proposed [1][7]. The XML tags and links marked in the SCM items can be collected and used to calculate them. We also suggest to use counts from constructs in UML diagrams to estimate Object-Oriented complexity. Some of OO complexity metrics and the related XML links/tags for their measurement are listed below:

(1) Project complexity
- Numbers of use cases (diagram tag)
- Average numbers of scenarios (scenario tag)

(2) Class complexity
1. Numbers of the messages in sequence or collaboration diagrams (*message* links)
2. Numbers of tracing links (*tracing* links)
3. Class Fan out (*use* links)
4. Numbers of Classes, Numbers of Attributes, Numbers of Methods (diagram tags)
5. Numbers of Children (NOC) (*inheritance* links).
6. Weighted methods per class (WMC) (method tags with assigned weights)
7. Depth of inheritance tree (DIT) (*inheritance* links)

These metrics present different viewpoints of complexity. Independent ones can be combined using weighted summation approach to form general complexity metrics.

4.2. Release management decisions

Our tools utilize Bayesian Belief Networks to support the decision making process of release management. BBNs provide modeling of uncertainties and support repeated evaluation given updated information. The data from configuration control and configuration verification can be plugged in for release decision. The BBN diagram for release decision is presented in Figure 7.

In Figure 7 "Release decision" is affected by "strategic issues" and "new version's product quality" nodes. While, "Strategic issues" is further influenced by the following factors:
- Time span from last release
- Market competition
- Portion of significant changes
- Company policy

On the other hand, the factor "New version's product quality" is further determined by the following:
- Maintainer's capability
- Number of outstanding changes per KLOC (K Lines of Code)
- Quality of the original version
- amount of detected defects

Among them, the "amount of detected defects" is in turn influenced by
- --Portion of significant changes
- --Number of the average abnormal reports per KLOC
- --Number of the average negative review comments per KLOC

Of all the nodes in Figure 7, we notice that values of the following three nodes are certain and can be obtained from tags.

(1) Number of the average abnormal reports per KLOC.
(2) Number of the average negative review comments per KLOC.
(3) Number of outstanding change per KLOC.

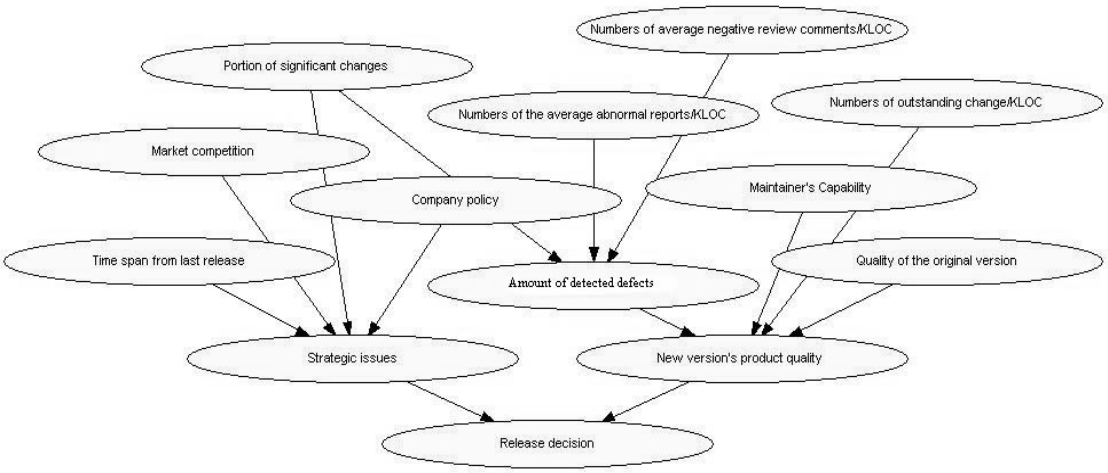

Figure 7. Release decision model

Since most of the factors in Figure 7 are uncertain, BBN estimation, as indicated above, can be used to support release decision. The above three certain data are first mapped by experts into numbers ranged from 0 to 1, and then they can be converted into probabilities of BBN nodes using fuzzy triangular functions shown in Figure 8. Take 0.6 for example, it will be mapped to the following probabilities: Low=0, mid= 0.8, high=0.2.

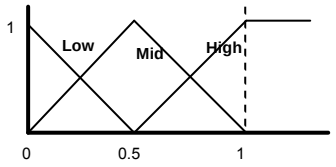

Figure 8 Fuzzy triangular function

. Figure 9 is a sample scenario where strategic issues node is highly favor release and the new version's product quality is estimated as high. This scenario is calculated using Hugin [2], a BBN's tool.

4.3. Maintainability prediction

We can also use information extracted from XML tags to predict software maintainability. Maintainability may be reflected through the following factors:

(1) Number of new/removed/corrected modules
(2) Number of revisions
(3) Average time required for impact analysis
(4) Number of outstanding change requests
(5) Average time taken to implement a change request
(6) Maintainer's capability
(7) Quality of the previous version

Items (3) to (5) are suggested by Sommerville [7]. Our tool can extract values of the first five items from XML tags and links. The last two factors are uncertain. Since the numbers of certain items are more than the uncertain ones, we use a hybrid approach to combine BBNs with weighted summation.

First, the uncertain factors, i.e., maintainer's capability and previous version's quality, are first expanded into BBNs shown in Figures 10 and 11. Once we have estimated probabilities of the "maintainer's capability" and the "quality of the previous version" using these BBNs, we can change their probabilities to real numbers using weighted summation. Finally, the values of these two factors are combined with all those of the other factors (items 1-5) using weighted summation to calculate the maintainability.

5. Evaluation and Conclusion

This paper presented an XML-based SCM environment along with data collection through XML tags and links for SCM related metrics. We have used a set of life cycle documents of an elevator simulator as a case study of our constructed SCM environment. Tests showed that the constructed tools could satisfactorily assist SCM tasks [9]. We have also tested the proposed metrics using extreme good and bad conditions. They all yield correct results [9].

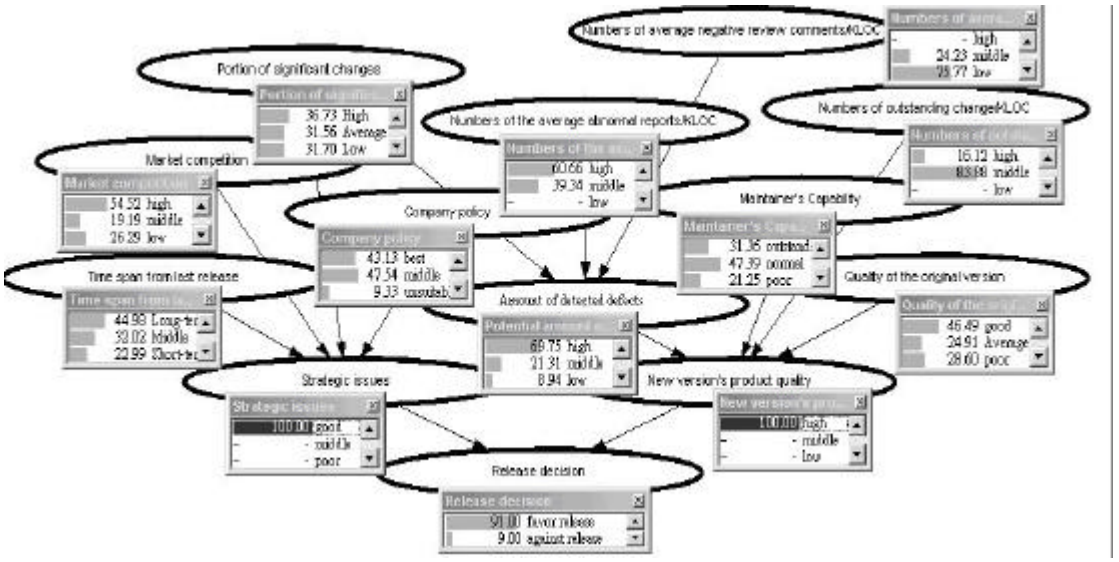

Figure 9. Favor release scenario

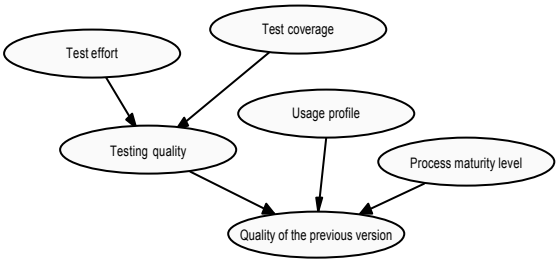

Figure 10. Quality of the previous version

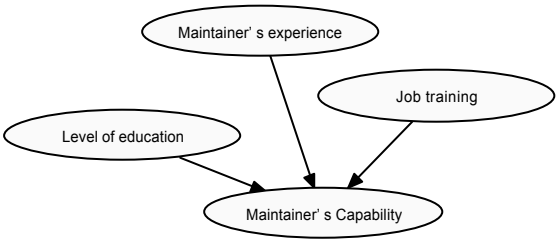

Fig. 11. Maintainer's capability

In summary, the major idea in the research was to use XML to express content semantics as well as the critical relationships among contents of configuration items. Thus, SCM tools can accurately and effectively retrieve related information to support major software configuration management activities, such as status accounting, impact analysis in change control, and configuration verification. Furthermore, the XML tags and links can be collected to provide metrics calculation, such as software complexity, maintainability, and release decision. The XML approach seems to be a perfect match for SCM tools and metrics.

References

[1] S. R. Chidamber and C. F. Kemerer, "A Metrics Suite for Object-Oriented Design," IEEE Transactions on Software Engineering, vol. 20, no. 6, Jun. 1994, pp. 476-493

[2] HUGIN EXPERT, http://www.hugin.com

[3] IBM Rational software, http://www.rational.com/

[4] "IEEE Guide to Software Configuration Management," ANSI/IEEE Standard 1042-1987, September 1987.

[5] F. V. Jensen, *An introduction to Bayesian networks*, UCL Press, London; 178 pages, 1996.

[6] MERANT, http://www.merant.com

[7] I. Sommerville, *Software Engineering*, 5th Edition

[8] Visible Systems, http://www.visible.com/

[9] Y. Weng, "XML-based Software Configuration Environment," master's thesis, Computer Science and Engineering Dept., Yuan-Ze University, July, 2004.