

Title	共同ソフトウェア開発における成果物に関する情報の管理方式
Author(s)	押目, 晴義
Citation	
Issue Date	2002-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/1562
Rights	
Description	Supervisor:落水 浩一郎, 情報科学研究科, 修士

Supporting Configuration Management of Artifacts in Cooperative Software Development

Haruyoshi Oshime (010027)

School of Information Science,
Japan Advanced Institute of Science and Technology

February 15, 2002

Keywords: software process model, software configuration management, XML, team model, modification thread candidate.

1 Background and Purpose

Recently, softwares are often developed in cooperative work. In cooperative software development, there are many developers with various roles and they create/modify many software artifacts such as UML diagrams. In such a case, it is very important to manage information about artifacts. It is desirable that information of an artifact contains not only description or diagram of the artifact but also so called configuration information about it, such as the reason why it is created, the history of the modifications for it, or dependence relation with another artifact. Using such a configuration information, we can support for example reuse of an artifact, keeping consistency of it, and to understand the process of developing it. Therefore, it is thought that a modification of artifact and a new artifact can be easily created. However, known techniques are not enough to achieve the above goal. For example, ready-made database schemes are not suitable, because they can deal with only the final software products. HTML is also not enough, because any HTML document can contain links to other documents besides the description of the document. So, we try to use a fashionable language to describe documents, XML. Using tags in XML, we can describe documents with various added values. There are some works on configuration management using XML. However, to our knowledge, they do not deal with "to know the range of influence of a modification for an artifact."

So, in this thesis, we propose the model to achieve the above goal using XML. In this model, information on dependence relation of an artifact is described in a tag in the XML document for the artifact. The range of influence of a modification can be computed by tracing this tag information, therefore we can obtain the modification threads by using this model. Furthermore, using the model, we try to give the method to analyze "interaction" among many modification threads. Interaction contains such as a race which occurs in a modification of one artifact belonging in some threads. Such a situation often occurs in a cooperative software development.

2 The Configuration Management Model

The configuration management model consists of the following two.

- (1) The Team Model, proposed by Ochimizu
- (2) The Modification Thread Candidates

The Team Model is a software process model which shows relations among roles, artifacts, communications, and so on. In this model, each role has its work space, and each space contains artifacts. Between two roles, a communication path can exist. Communication between roles may occur using this path, and it may cause that some artifacts are owned by both roles in common. As the result, a change of the range of each work space may occur.

The modification thread candidates C for an artifact a of a Team Model T is defined as follows. Assume that a team model T has a set of artifacts A . First, we the dependence graph $G = (A, B)$ for T . The node set A is the above set A of artifacts. The set B of arcs is the dependence relation among the artifacts in A . Note that G is a directed tree. The modification thread candidates C for an artifact $a \in A$ of T is the subtree of G with the root node a .

3 The proposed management method is as follows

We propose the following method to support modification of artifacts in cooperative software development. Also assume that a team model T has a set of artifacts A .

1. For each artifact in A , compute the modification thread candidates C
2. For each artifact (node) in C , check the work space where the node belongs and the role which own this work space.
3. For the roles detected in the above, find communication paths between two roles of them. Furthermore, check the frequency of communications for each path, by using other configuration information.
4. For all modification thread candidates, find all conflicts which may occur in modification processes. This is achieved by finding nodes belonging to more than one candidate.
5. Considering the conflicts detected in the above, give the schedule for modifying artifacts. The schedule uses the "lock" for some threads, and the locks are due to some configuration information.

Considering the conflicts detected in the above, give the schedule for modifying artifacts. The schedule uses the "lock" for some threads, and the locks are due to some configuration information.

4 Management System and Evaluation

In this thesis work, the methods 1,4 and 5 are implemented. The evaluation is based on this system. We evaluate the method under various conditions on the number of the artifacts, the number of the artifacts to modify, the number of the dependence relations and the strength of each dependence relation. The system simulates the modification process under each condition. We define the performance of the system by the following two point of view: the sum of the length of all threads to modify and the number of the times modifying artifacts. According to such conditions, the range of the condition for locking is determined.

The result of the evaluation shows that the system is effective when the number of the artifacts is sufficiently large, compared with the number of the artifacts to modify and the number of the dependence relations. The system is also effective when each strength of the dependence relation is strong. The result of the evaluation also shows that the performance of the system depends on the selection of the condition for locking in the range.

The use of the system according to the situation which the user demands is possible by changing the threshold. The situation by which either distance number or update frequencies is valued and the situation by which both balances are valued, etc. Moreover, it seems that the system becomes more effective by enhancing the element to decide the threshold.

Therefore, when seeing from the viewpoint of presentation of various possibilities to need, the management system is effective enough in the modification work in the cooperative software development.

5 Summary and Future Works

In this paper, we examined various artifacts created in the development process in the cooperative software development about the management method of information of those configurations. Concretely, we proposed the management model by which information to modify the artifact which added the modification thread candidate to the team model was able to be managed, as a method of the effective of giving the efficiency of the change work. Moreover, we proposed the management method which showed the possibility of information necessary for the change in the artifact by using the management model. In addition, we implemented the prototype of the management system based on the management method, and reported on the result of doing the evaluation experiment by the simulation of the management system.

The result of this research by the management model is summarized on the following.

- The modification order, update frequency and communications frequency which modifications and depends for the artifact can be guessed.
- The process situation of the change work can be understood, and the modification work which apply to the situation can be proceeded by the best schedule.

The future works is as follows.

- We will evaluate the management system which adds 2 and 3 of the management method in an actual cooperative software development.
- We will examine the tag to enhance the modification cost of the artifact and the combination.