CoFM: An Environment for Collaborative Feature Modeling

Li Yi, Haiyan Zhao, Wei Zhang, Zhi Jin
Institute of Software, School of EECS, Peking University
Key Laboratory of High Confidence Software Technology, Ministry of Education of China
Beijing, China
{yili07, zhhy, zhangw, zhijin}@sei.pku.edu.cn

***Abstract*—Feature models provide an effective way to capture commonality and variability in a specific domain. Constructing a feature model needs a systematic review of existing software artifacts in a domain and is always a collaboration-intensive activity. However, existing feature modeling methods and tools lack explicit support of such collaborations. In this paper, we present an environment for feature modeling that promotes the collaboration between stakeholders as the basis of creating and evolving a feature model. We present concepts, methods, and a tool to show the feasibility of constructing feature models collaboratively, as well as how to integrate this environment with traditional feature modeling methods.**

Keywords-feature model; collaboration

I. INTRODUCTION

Feature models provide an effective way to organize and reuse software requirements in a specific domain. A feature model consists of a set of features and dependencies between these features, and it is an abstraction of commonalities and variabilities among a family of applications inside a domain.

The construction of feature models is a collaboration-intensive activity. For example, FODA [1] method suggests that domain analysts (i.e. constructors of feature models) intensively communicate with domain experts, users, and other stakeholders to gather information about the domain. The reason for collaboration is obvious: most domains are complex and often evolve frequently, thus it is usually impossible for one person to obtain a comprehensive understanding of a domain without sharing knowledge with others.

The stakeholders' collaboration faces many problems. First, the natural language used for communication is often ambiguous, hardly expresses precise semantics of features or feature dependencies. Second, a stakeholder cannot track everyone's activity, nor every detail of a feature model, so risks duplicating or conflicting with others' work. Furthermore, it takes a lot of time and effort to resolve the conflicts between stakeholders. The above problems make stakeholders' collaboration a bottleneck of the construction of feature models.

However, few of existing feature modeling tools have provided explicit support for efficient collaboration among stakeholders. They just leave the work to domain analysts, and it leads to two consequences. First, the effectiveness of collaboration highly depends on domain analysts' knowledge and experience so that it is often unsatisfying. Second, it imposes heavy workload on domain analysts to guide the collaboration, particularly the resolution of conflicts. In short, construction of feature models is a time-consuming and error-prone task. We believe that the situation may be alleviated by integrating collaboration support with feature modeling activity in a more systematic way.

In this paper, we present an environment developed for collaborative feature modeling (CoFM), by explicitly making collaboration a central activity in constructing feature models.

II. THE CONCEPTUAL MODEL OF COFM

Fig. 1 gives the conceptual model of CoFM. The *persons* collaboratively construct a shared *feature model.* There are two kinds of knowledge that *stimulate* the persons perform *modeling activities*. One kind of knowledge, the *external* knowledge, comes from the outside. For example, textbooks, documents, source code, personal experience and software itself belong to this kind of knowledge [1] [2]. The other kind of knowledge, *internal* knowledge, is the shared feature model itself. Since the feature model reflects opinions from many people, one can therefore learn from others. Individual modeling activities are mashed-up into the shared feature model, and meanwhile, reshape *personal view* for each person. Personal views can be reused outside the environment.

The essential part of the conceptual model is *modeling activities.* Intuitively, people can *create* elements of feature models, including features, dependencies and attributes of features. However, if there are no confines of creations, the environment might be filled with poor elements and the overall quality of the shared feature model deteriorates. So

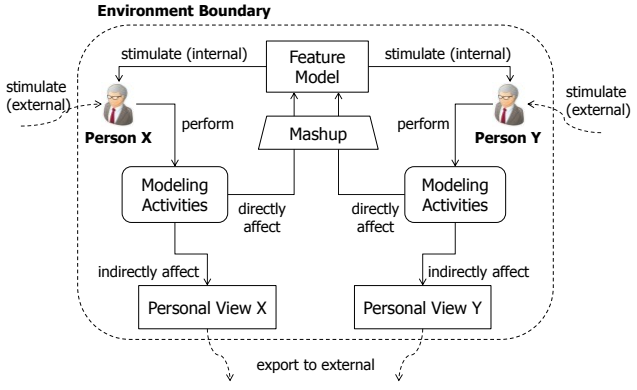

Figure 1. The conceptual model of CoFM.

we introduce two operations to let people *evaluate* others' work, that is, *select* or *deny* an element. People can select a good/useful element into their personal view, and deny a bad/useless element to keep the element away from their personal view. Analysis on the select/deny operations can distinguish elements of high quality from those of low quality, and also distinguish core contributors from irresponsible or even malicious users.

The personal view of a person consists of elements created or selected by the person. In other words, a personal view can be treated as "a feature model that only reflects a specific person's opinions about a domain". Since such opinions are either come from the person himself (by *creating*) or learned from others (by *selecting*), the personal view is conformed to a feature model constructed with existing, non-collaborative feature modeling methods (in which a feature model is a domain analyst's personal view). However, there are two obvious differences. First, the CoFM environment is opened to virtually everyone so the potential amount of stakeholders is far more than non-collaborative methods. It means that CoFM can be very useful for real feature models that often contain thousands of features and dependencies. Second, all stakeholders, not only domain analysts, interact with the feature model directly and express their opinions in terms of features and dependencies directly. This is more efficient than existing methods in which domain analysts have to transform others' opinions expressed in natural language into concepts in feature models.

III. THE TOOL

We have developed a web-based tool to implement the CoFM conceptual model. The latest stable version can be accessed via http://cofm.seforge.org/cofm/.

A. Modeling Activity Support

Fig. 2 shows a screenshot of the main modeling UI of CoFM. The shared feature model and current user's personal view are displayed in the left panel. When a feature is clicked, its attributes and related dependencies are shown in the top-right panel, as well as evaluation operations (select and deny) for this feature. Other information such as history and comments are displayed in the bottom-right panel.

The mashup of individual modeling activities (see Fig. 1) is implemented as follows. Once a modeling activity (create, select or deny) is performed, it is immediately uploaded to the central server where the shared feature model is stored. Then we try to apply the activity to the feature model, that is, if the activity's pre-condition is satisfied, it is successfully applied, otherwise it is failed. An applied activity is then sent to all other people therefore it becomes visible to others. A failed activity is simply abandoned. An activity may fail due to concurrent activities on the same element by multiple persons. For example, suppose a user denies a feature, and when this activity is applied, the feature is removed since all users have denied it now. However, it takes a little time (network delay) to make this activity visible to a user, say, *X*. If *X* selects the feature during the network delay, *X*'s activity will be uploaded to server and then failed because the feature has already been removed at the server.

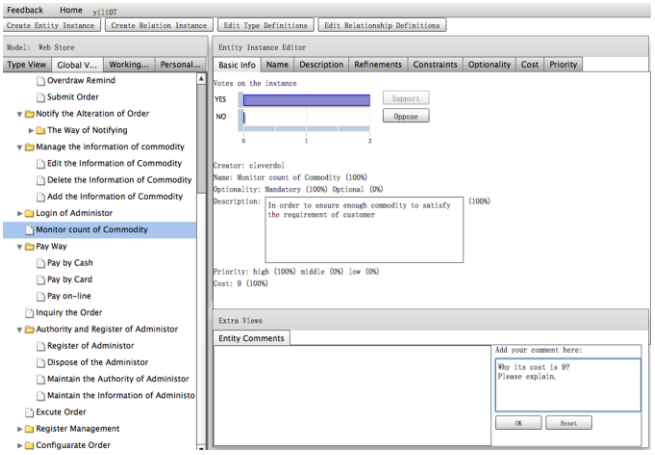

Figure 2. Main modeling UI of CoFM tool.

B. Customizable Feature Attributes

An essential task of feature modeling is to describe what a feature is, that is, explicate attributes of a feature. In an open environment like CoFM, we cannot predict how the stakeholders will describe a feature (we do not even know who stakeholders are). For example, a programmer may be interested in *ease of implementation* of a feature, while a customer may focus on *importance* of a feature. Our solution is to make attributes of features *customizable*. We provide name and description as default attributes, and also allow users freely add new attributes. The attributes must be one of these types: *string, text, number, enumeration,* and *pointer* (a reference to another feature).

IV. CONCLUSIONS

We have introduced CoFM, an environment for collaborative feature modeling. Our major contribution is that CoFM is the first feature modeling environment that places collaboration between stakeholders at the heart of construction of feature models, and it is compatible to existing feature modeling methods by introducing personal views. We also developed a web-based tool to implement the environment.

ACKNOWLEDGMENT

This work is supported by the National Basic Research Program of China (973) (Grant No. 2009CB320701), Key Project of National Natural Science Foundation of China (Grant No. 90818026), National Natural Science Foundation of China (Grant No. 60873059), and the National Key Technology R&D Program (Grant No.2008BAH32B02).

REFERENCES

[1] Kang, K. C., Cohen, C., Hess, J., Nowak, W., Peterson, S. Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University, 1990.

[2] Zhang, W., Mei, H., Zhao, H. Y. A feature-oriented approach to modeling requirements dependencies. In Proc. of the 13th IEEE Intl. Conf. on Requirments Engineering. RE '05. 273-282.