Real-time Collaborative Modeling

Muzaffar Artikov

1 Introduction

Modeling is a key activity in conceptual design and systems design, and a crucial step in the development of simulation models [?, ?, ?]. There is broad agreement that is important to involve various experts, stakeholders and users in a development cycle. If users are not involved in systems analysis tasks, their problems, solutions and ideas are difficult to communicate with the analyst. This often results in poor requirements definition, which is the leading cause for failed IT projects [?]. Thus, it is important to organize systems modeling process collaboratively between analysts, developers, stakeholders and users.

Collaborative modeling is the process of joint creation of a shared graphical representation of a system. While there is means to verbally explain models, such as metaphors, a graphical representation is often more effective. ("A picture tells more than a thousand words"). To use graphical representations as a basis for discussion, it would be useful if all the stakeholders can be actively engaged in the construction and modification of such models [?].

Collaboration means all members establish goal together and solve problems in a cooperative manner to reach common objective [?]. Collaboration is usually divided into two types: data-based collaboration and reciprocation-based collaboration. The former, such as data transmission and design technique exchange, narrates mainly the sharing of data and knowledge through the integration of artificial intelligence and database technology. While the latter discusses the situation of real-time and synchronous operations between the participants in a collaborative process, such as real-time design [?]. The collaborative software design assembles many relative personnel who simultaneously participate in the software development process, including designer, developer, project manager, manufacturer, supplier and marketer. Members from different locations can communicate and discuss issues together to concurrently carry out the software design and modification via the network that makes design results more in accordance with the consumer's requirements [?].

Software engineering involves teamwork and communication of many kinds. Specific examples include:

- In agile processes, such as eXtreme Programming, pair programming requires very close collaboration focused on the same artifact. In collaborative software engineering (CSE), the pair members need not be spatially co-located.

- Development activities such as analysis, design, testing and coding maybe carried out by different combinations of individuals. CSE-mediated discussions are potentially a valuable way for effective communication and feedback between and within these groups.

- When correcting defects, team members may consult former team members, currently assigned to other projects, in order to determine the rationale for some design feature which has subsequently been identified as problematic.

- Refactoring often involves relatively minor but widespread changes. Users who are kept informed of such activity are able to avoid potential contacts [**?**].

Various collaboration models are normally defined by time and location of collaboration and operation of collaboration such as close coupled and loosely coupled [**?**]; face to face collaboration, synchronous distributed collaboration, asynchronous collaboration, asynchronous distributed collaboration [**?**]; mutual collaboration, exclusive collaboration, dictator collaboration [**?**]. The strength of face to face collaboration, synchronous distributed collaboration, close coupled collaboration and mutual collaboration is to provide on-line compiling and real-time operating functions for participants to reduce process duration. And its weakness is the difficulty of reviewing and auditing personal performance in an effective way. On the other hand, the strength of loosely coupled collaboration, asynchronous collaboration, asynchronous distributed collaboration and exclusive collaboration is no need to gather all participants to accomplish the work with a specific location at one time. The weakness of this kind of collaboration is the work breakdown structure may cause potential impacts on task completion due to individual delay [**?**, **?**, **?**].

For modeling tasks, Unified Modeling Language (UML) or UML based languages such as SoaML, SysML, etc. are usually used in software modeling. The UML is a general-purpose, developmental, modeling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system [**?**]. There are 9 types of diagrams to describe the whole system in UML.

Real-time collaborative modeling envisages a joint creation of models from several geographical places by using several computer machines in real-time. Recent advances in software engineering allow collaborators to synchronously edit artifacts. From an engineering perspective, adding real-time, multi-user collaboration to single-user applications is a challenging task as it requires the implementation of features such as conflict resolution as well as propagation and visualization of updates in real-time [**?**].

2 Motivation

Software modeling require team efforts in which group of experts from various fields work together. Close collaboration between them, especially, in a real-time environment, is important to the success of a development work. This is because software systems are getting larger and more complex, and developing these systems requires efforts from multiple designers, who may be geographically dispersed. Consequently, tools must exist to allow designers to collaborate effectively in each stage of a software development process.

The evolution and maintenance of complex software systems require the collaboration of several designers. The importance of real-time collaboration can be seen in many cases. For example, in case of learning, junior software designers can learn faster from experienced senior designers in developing process.

Therefore, real-time collaborative modeling system must be developed that will provide real-time synchronization over the network if the network is available, otherwise, all the additions, changes and deletions done by user should be stored on local machine and should submit these changes when the network is connected and available. Storing on local machine should be as in version control systems, i.e. there should be an opportunity to head back to evolution stages. Thereafter, when network is available changes made on the models should be sent to the repository. And should be found ways to resolve conflicts at this case.

3 State of Art

Collaborative modeling as a field of both practice and research has developed over the last decades. Within the field of system dynamics, modelers started to involve client groups in the modeling process since the late 1970s [**?**]. Since that time, various other modeling schools have adopted the notion of collaborative modeling and found approaches to involve stakeholders in their own modeling efforts [**?**].

Young Bang et al. on their article [**?**] represented CoDesign – collaborative software modeling framework. CoDesign's main contribution is an extensible conflict detection framework for collaborative modeling. CoDesign utilizes an event-based architecture in which highly-decoupled Components-different instances of CoDesign - exchange messages via implicit invocation, allowing flexible system composition and adaptation. In the article modeling-level conflicts categorized into three classes based on the rules that the system modeling events violate: synchronization, syntactic and semantic conflicts.

Kuryazov et al. on their article [**?**] represented Delta Operations Language (DOL), which is utilized to model difference representations. DOL is a set of domain specific languages to model difference representation in terms of delta language operations. In order to derive a specific DOL for a specific modeling language, the meta-model of a modeling language is required. A DOL Generator used to generate a specific DOL for a certain modeling language using the meta-model. Then, a specific DOL is fully capable of representing all differences between subsequent versions of the instance models in terms of operation-based DOL in modeling deltas. The operations in modeling deltas are referred to as Delta Operations. DOL aims at supporting several DOL-services, which can access and reuse delta operations. These operative services make the DOL-based modeling deltas quite handy in various application areas and enable application areas to utilize the DOL-based modeling deltas. A specific DOL is derived for a specific modeling language and several DOL-services. DOL-services provide means to manage DOL-based deltas. Services relying on representation of the DOL approach are applied to model versioning and model history analysis, using state-based difference calculation [**?**]. DOL can be utilized as a strong basis for organizing the collaborative modelling process. The main goals of representing DOL are: implementation of understandable high-level modeling language and organizing collaborative modeling process on basis of it. But, as authors mentioned, DOL-services list should be extended by adding a runtime operation recording service, difference merger service and synchronizer service for organize real-time collaborative modeling process based on DOL.

4 Research Question

This is the research question section

5 Conclusion

This is conclusion section

References

[1] D. Kuryazov, A. Solsbach, and A. Winter. Towards versioning systainability reports. Number 05/2013, Oldenburg, 2013. Carl von Ossietzky University, Oldenburg, Software-Engineering.

[2] D. Kuryazov and A. Winter. Representing model differences by delta operations. In *EDOC Workshops*, pages 211–220, 2014.