

An Agent-Based Design Software : A CSCW Model for Software Design Tool

P. Bhattarakosol

Department of Mathematics,
Faculty of Science
Chulalongkorn University,
Bangkok, Thailand 10330

W. Pacharoen

Department of Mathematics,
Statistics, and Computer Science
Faculty of Science,
Ubon Ratchathani University,
Ubon Ratchathani, Thailand 34190

L. Preechaveerakul

Department of Computer Science,
Faculty of Science
Prince of Songkla University
Hat Yai, Thailand 90110

***Abstract** - A technique called as Computer Supported Cooperative Work or CSCW is a technique to support communications among people. This research presents the architecture of an agent-based software that is applied as the analysis and design tool during the analysis and design processes. This tool allows communications among development team members, and automatically records all conversations related to the designed diagrams. The communication modules perform both synchronize and asynchronize mechanisms to gain the efficiency and security while the communication occurs. The concept of implementing separated databases for different class of users has been proposed as a method for data security control. Various modules such as Discussion Module (DM) have been implemented to enhance the quality assurance for software development processes under the CSCW environment. The result of this proposed architecture is recorded details that can be used as a quality control document and a maintenance document.*

Keywords: Agent-based architecture, communication channel, computer supported cooperative work, understandability.

1 Introduction

Software development is a task that can be succeeded if and only if there are good communications among the development team and stakeholders. Thus, the formal meetings among the stakeholders and the development team must be arranged during the development period. The expected result of the meeting is the qualified software delivered to the clients. However, arranging each formal meeting with clients is not easy as wish because of the time constraint of the clients.

Furthermore, the software process must be certified under some specific rule such as CMMI. Therefore, every step of the development process must be documented and reported in a proper format so that the software can be maintained after delivered. Important details that are needed to be record in the hand-in documents are the problem-reason-solution of each designed part and

implementation of software. Unfortunately, these details are hardly be record, and, most of the time, the important reason for the solution were left out.

CSCW has the full name as Computer Supported Cooperative Work that is stated in the year 1984 in the workshop arranged by Iren Grief of MIT and Paul Cashman from Digital Equipment Corporation [5]. CSCW was implemented in many environments to support different tasks such as education, software design, telemedicine, e-commerce, etc. In this research we applied the concepts of CSCW for the software development process so that the qualified software will be produced

The development of CSCW in the early age is that developers were focused to all participants who work on the same network. Therefore, Jonathan proposed CSCW as a forum of all participants [4]. When Internet becomes popular for network users, the idea of implementing CSCW over Internet was proposed in [3]. Thus, people from different places around the world can communicate and share data at the same time using the available network.

In the year 1997, Chin-Hwa et al. proposed the system architecture of CSCW that consisted of two important subsystems: the session management, and the session processing subsystems [9]. This architecture was also implemented based on the multimedia technology and the communication over the network to create the real cooperative work environment. Additionally, this architecture had implemented a database to store the important information for the cooperative tasks

The basic mechanism of CSCW was implemented using the synchronous process until Sheng and Andy proposed the first asynchronous CSCW in the year 1998 [10]. However, the result of using new technique does not have any differences from the previous mechanism.

A new CSCW architecture called Viceroy was proposed by Xuanming in 1999. This new invented CSCW supports the model of graphic presentation. Thus, people can clearly understand the discussion information based on the presented models. The Viceroy CSCW also protects the inconsistency of the presented model to users.

Therefore, all users will receive the same information and models at the same time.

Since object-oriented technology had been introduced and implemented widely, Yahiki et al. [7] applied this technique to implement a CSCW that can be modified to serve various working environments. However, the developed CSCW does not really support the cooperative work of people who have different experience [2]. Therefore a Personal Assistant (PA) module for knowledge management was added to be a part of the CSCW by Fabricio and Jean-Paul. This PA supports all workers who confused with information that they received.

Although the objective of CSCW is to support people from different places to be able to work together, but each person still needs a private area to perform their tasks. Therefore, the concept of granting a private area for users was proposed in [8]. Using this private area helps people to be more productive than fully share the information all the time.

Liu Hong and Cui Wei proposed a multi-agent collaborative software design environment which can design hardware components [6]. This proposed software supports the design architecture in three dimensions.

Since there are various methods have been proposed to support the cooperative work using computers, there are many defects to be considered such as the security for data using in the distributed system, or the design concept that should be stored as references during the maintenance periods, etc.

This research proposes the architecture of an agent-based software that is applied as a tool for drawing diagrams and also supports the communication among team members to be able to discuss problems and define their solutions. Additionally, all conversations among the team members are completely stored in a database. So, the concepts and reasons for each solution can be reviewed during the maintenance process.

2 System Architecture

It is the fact that the good software design can be achieved from the frequent meetings. However, it is not easy to set a meeting date and time for a large teamwork in a short period of time. Therefore, CSCW technology was introduced in order to connect all software designers and stakeholders together on one system. In this paper, a system architecture that all users can see each other as what-you-see-is-what-I-see (WYSIWIS) has been proposed.

Considering the design process, the users in the CSCW system can be classified into two categories: the person who draws diagrams (will be called as diagram designers or DD), and the person who verifies the diagrams (will be called as the diagram verifier or DV). Each category has different and independent roles. The DD is the person who is responsible for drawing and

designing diagrams in the system development process, both in analysis and design phases. During the analysis phase, analysts are DD. During the design phase, the system/software designers are DD. Therefore the DD in the analysis phase can be different from DD in the design phase.

On the other hand, DVs are members of the development team who participate in the diagram verification. The conversation among DD and DVs that affects to the changes of diagrams is important and must be recorded to achieve the software quality assurance.

2.1 Roles of DDs and DVs

Since the privacy and security are two important issues in distributed applications and information, therefore, in this research, the responsibilities of DD and DV are separated in order to obtain these issues.

The identify functions of DD are as follow:

1. DD is the person who responsible for deriving diagrams during analysis and design phases. Furthermore, each diagram belongs to a particular DD only. So the security of the diagrams can be protected.
2. The authority to accept or reject connections to DVs belongs to the DD only. In the situation that a DD is not ready to connect to any DVs according to some important reason, thus DD has authorized to reject the request for connection from other DVs.
3. DD has authority to propagate the designed diagram to DVs, one diagram at a time. In order to avoid the diagram confusion of DVs, only one diagram will be sent for each transmission.
4. DD has authority to disconnect the link to a particular DV. After the end of the conversation or disconnection between DD and a DV occurs, the DD can terminate the link to DV as necessary.
5. DD has authority to make negotiations with any DVs, and every negotiation must be recorded and seen in the modified diagrams. Additionally, all negotiations can be viewed by any DVs for their understanding and comments.
6. All discussions without certain conclusions will not be counted as the agreement between DD and DV, but these discussions will be stored in the diagram repository system for further usages.

A DD can propagate the discussion conversations, based on the propagated diagram, to all DVs. Since the main authority to get access to the diagram is granted to DD, therefore the roles of a DV are stated below:

1. DV has right to request the connection to a DD's system whenever a DD is logged on.
2. DV has right to discard the connection to a DD's system.

3. DV has right to send the request for discussion to DD when needed.
4. DV has right to make negotiations related to the presenting diagram with a DD.
5. DV has no right to update or change all received diagrams from DDs in order to maintain diagram correctness and security.

Although the responsibilities of both partners, DD and DVs, are dependent and mainly relied on DD's functions, the consistency of the information that all DD and DVs is still maintained.

2.2 Design Components

According to the functions defined in Section 2.1, the system will be classified into two agents: the design agent, and the verifier agent as shown in Figure 1.

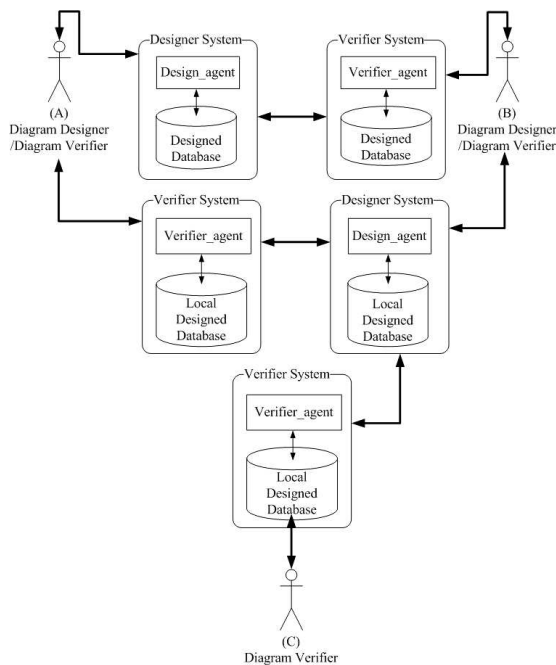


Figure 1. System Functions of the Agent-Based Design System.

Since the functions are clearly separated, therefore on one machine both agents can be installed. Thus, a team member can be a diagram designer and a diagram verifier in the same time.

Referring to Figure 1, there are 3 persons who participate in the design process, A, B, and C. According to the A's system, there are two agents installed so that A is the diagram designer for diagram-A, and in the same time A is the diagram verifier for diagram-B. On the other system, B's system, B is the diagram verifier for the diagram-A, and also be the diagram designer for the

diagram-B. Additionally, diagram-B is also verified by C but C is not the diagram designer for any diagrams in the system.

In order to obtain the sufficient design system that can guarantee the quality of the software design process, the functions that the agent-based design software are stated are as follow.

- The software must manage the connections among DDs and DVs, where one person can play both roles, without communication confusion.
- The software must store diagrams, discussions, and negotiations among development team whereas all details are consistent and up-to-date to each other.
- The software must maintain privacy and security of the designed diagrams repository.

From Figure 1, there are 2 main subsystems as same as 2 roles that had mentioned previously. One subsystem is the designer system, and another subsystem is the verifier system. The full architecture of the proposed software is presented in Figure 2.

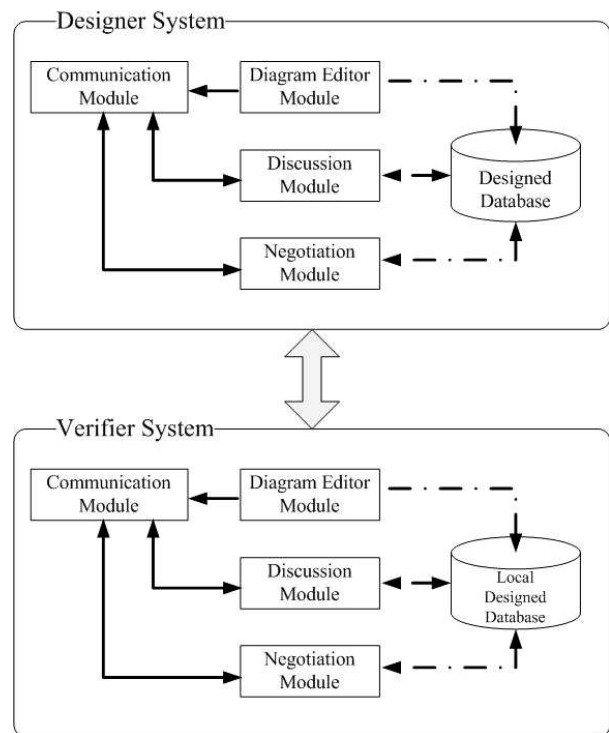


Figure 2. The Agent-Based Design System Modules

From Figure 2, the subsystem at the Designer System (DeS) is composed of 4 main modules: Diagram Editor Module (DEM), Discussion Module (DM), Negotiation Module (NM), and Communication Module (CM). Additionally, a database is implemented to store all

diagrams, discussions, and negotiation, called as Designed Database (DDB).

On the other hand, the Verifier System (VeS) is also composed of 4 main modules which are similar to the designer system except the Diagram Presentation Module (DPM). The local designed database (LDDB) acts as a duplicated database of DDB located at the designer's system.

Diagram Editor Module (DEM)

This module is responsible for creating and updating diagrams with respect to a project. Since a project consists of many diagrams from analysis and design phases, the data such as project name, project-id, diagram-id, date and time of creating or editing the diagrams, the designer's name, including with the drawn diagram will be entered to DDB. All data in DDB are used to maintain consistency of data from the other 2 modules: DM, and NM.

Diagram Presentation Module (DPM)

Every verifier must set its system IP using GUI of VeS. This information will be used as the indicator of each verifier when a connection is requested or granted.

After the connection is granted, DEM sends a diagram through the communication module and presented to the verifier by DPM. The verifier is not authorized to modify the received diagram. Therefore, the function of DPM is the one-way communication, a receiver, as same as DEM's function, a sender.

The most important function of DPM is to update details of LDDB so that LDDB will have the same content as DDB at DeS, and the consistency of the information in this distributed process can be obtained. DPM is allowed to retrieve data in DDB, not only the diagrams, but also all details of discussions and negotiations. Thus, the verifier can understand all the changes of the entire system.

Discussion Module (DM)

This module is responsible for all discussions that occur during the connection between the designer and verifiers. Since there can be multiple connections at a time, the participant's name will be stored as an additional information in DDB and LDDB. As the fact that the discussion is relied on the particular diagram, therefore the information in DDB are used to classify the discussion to the certain diagram.

Negotiation Module (NM)

The negotiation means an agreement that makes among a designer and a verifier to modify the presented diagram or an agreement from development team members. As same as DM, the negotiation is based on the presented diagram. Thus, information stored in DDB is

used to identify the diagram in which the negotiation has been stated.

According to functions of NM, every change with reasons of the change will be recorded in DDB and LDDB. Therefore, the content in DM will not be counted as the negotiation unless it is entered by NM.

Communication Module (CM)

This module is the most important part that creates connections among a designer and verifier(s). Although the main task of this CM is to deliver the connection for communication, but the role of CM at the designer's system is different from the verifier's system.

One function of the CM of DeS is to verify the request from the verifier system, whether the connection can be granted or can be ignored. If there is a request arises to the designer system, the request message will be verified by CM of the designer system before asking the permission from the designer; otherwise the request will be ignored and cascaded. If the designer wants to grant the connection, then the connection button on the GUI is pressed and the connection starts. The second function of DEM is to receive messages from the CM of a VeS and distributes to every verifier who connects to DeS in that time. Thus, the information on each VeS will be the same as DeS.

On the other hand, the CM of VeS has responsibility to send the request of connection to the designer's system, asking for connection. This message contains the IP address of the authorized system and the name of the verifier, and these information are used for identifying the right of connection.

Whenever the designer grants the connection for a request, all information in LDDB will be updated by DPM using a function of CM, then the presented diagram on DEM will be transmitted and presented by DPM at the DV's site.

As same as the diagram, the messages from DM or NM can be transmitted to both sites by CM whenever the sending button of DM or NM is pressed. The CM has no authority to retrieve data from DDB or LDDB.

By the conclusion, the functions of the CM of DeS are (1) verifies the request for connection (2) opens the connection (3) informs the communication situation (4) sends diagram or message to the connected CM (5) distributes received message from a CM to every connected CM. Whereas the main tasks of the CM of VeS are (1) sends the request for connection to DeS (2) receives (or transmits) data packets from (or to) the DeS (3) inform the communication's situation when it is completed or failed.

Since the synchronized and asynchronous processes are effective and suitable in different situations. Thus, CM will implement both synchronized and asynchronous

processes in different modules for handling different requirements.

Diagram Repositories

Since the security issue is a significant issue, and the verifiers should be able to review the design any time they want. Therefore, instead of installing only one database as proposed in [1, 7, 9], this architecture consists of 2 databases: a local database for a verifier, a local database for a designer. Thus, the verifier can review the design details stored in the local database at the verifier's site without retrieving data from the designer's site.

According to maintain the correctness of information of both DeS and VeS, a Designed DataBase (DDB) and a Local DDB (LDDB) are installed. The structure of these two databases is the same.

Both DDB and LDDB consist of 4 tables: system information (SI), project information (PI), figures (FG), and negotiation (NG). The details of each table are listed as follow.

- The SI table stores information of the VeS which are the verifier's name, and the system IP address that was entered to the system via GUI of DeS and VeS.
- The PI stores information of a project: the project identification, the project name, the designer IP address, the start date, the last update, the status (designer/verifier), the diagram version, and the negotiation version.
- The FG consists of the project identification, start date, the shape of the object (such as square, circle, etc.), the color-codes for red-green-blue, width, the position of object on the drawing sheet, and the version.
- The NG is used to store details of the negotiation based on the presented diagram. The contents in this table are the project identification, the start project date, the last update, the name of person who put the negotiation, details of the agreement, the status of the agreement (done or wait), acceptance status (this status is a confirmation flag when the verifier accepts the modification based on the negotiation), and version.

This software was implemented using C# under .NET framework where object-oriented design model can be implemented easily.

3 Conclusions

In order to develop qualified software, the good cooperation among development team is important. All discussions occur during the diagrams design are also

important because these discussions usually lead to the changes of each diagrams which is important information for maintaining software after software delivery, or sometimes for the reviewing of the development processes. Therefore, the objective of developing the agent-based CSCW is to support the development process mentioned above with short time and low cost of software investment.

According to the architecture of the agent-based CSCW, it consists of two important agents: the designer agent, and the verifier agent. These two agents have individual responsibilities and are installed on different computer systems. The designer agent is installed at the designer's site, individually. On the other hand, the verifier agent is installed at the working team computers. As the fact that designers are also in the working team so it is possible that the verifier agent is installed on the same site of the designer agent. However, not all designers need to participate in the other designers' designs, and not all members in the working team are designers. Thus, some members will use only the verifier agent while some members of the team will use both designer and verifier agents. Since DDB is installed at the designer agent and will be accessed by the owner of the diagrams only, so the consistency and integrity of the database can be maintained. On the other hand, the security of the data at the server agent, diagrams and DDB, will depend on the group policies including the security protection abilities of the operating system of the DD's computer.

The result of using the agent-based CSCW software to develop software will help the development team to communicate easily and quickly than the meetings in a room. Additionally, the cost for setting a meeting at a certain place will be eliminated because group discussions are able to manage while the development team can be at any places, at any time. The other advantage of using the agent-based CSCW is that all conversations that relate to diagrams will be recorded in DDB for referencing and maintaining software after the software is delivered. However, the only one disadvantage in using the agent-based CSCW is that the agent-based CSCW cannot force DDs to correct diagrams according to the agreements from their discussions. Therefore the agent-based CSCW will be useful and efficient if all DDs are honesty to their works and their agreements.

4 References

- [1] Xuanming Dong and K. R. Subramania. "Supporting cooperative work with Viceroy." In *Proceedings of the 3rd International Conference on Computational Intelligence and Multimedia Applications*, New Delhi, pp.437-441, Sept. 1999.
- [2] Fabricio Enembreck and Jean-Paul Barthe. "Personal assistant to improve CSCW." In *Proceedings of the 7th International Conference on Computer Supported*

Cooperative Work in Design, Rio de Janeiro, pp 329-335, Sept. 2002.

[3] Tom Gross and Roland Traunmuller. "Computer-supported cooperative work and the Internet." In *Proceedings of 7th IEEE International Workshop on Database and Expert Systems Applications*, Zurich , Switzerland, pp. 425-430, Sept. 1996.

[4] Jonathan Grudin. "The CSCW forum." In *Proceedings of the 26th Hawaii International Conference on System Sciences*, Hawaii, pp.51-58, Jan. 1993.

[5] Jonathan Grudin. "Computer-supported cooperative work: history and focus." *Computer*, 27(5), pp. 19-26, May 1994.

[6] Liu Hong and Cui Wei. "Supporting Design in an agent based collaborative environment." In *Proceedings of the 8th International Conference on Computer Supported Cooperative Work in Design*, vol.1, pp 348-355, May 2004.

[7] Yahiko Kambayashi, Kaoru Katayama and Mukesh Mohania. "Modifying CSCW environments dynamically for supporting virtual enterprises." In *Proceedings of 9th International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises*, Sydney, pp.80-87, Mar. 1999.

[8] Larry Korba. "Towards distributed privacy for CSCW." In *Proceedings of the Sixth International Conference on Computer Supported Cooperative Work in Design*, London, Ontario.,Canada, pp 95-101, Jul. 2001.

[9] Chin-Hwa Kuo, Timothy K. Shih, Hsi-Tsung Wang and Chun-Kai Juan. "A system architecture of CSCW." In *Proceedings of the 1997 International Conference on Intelligent Processing Systems*, Beijing, vol. 2, pp.1612-1615, Oct. 1997.

[10] Sheng F. Li and Andy Hopper. "A framework to integrate synchronous and asynchronous collaboration." In *Proceedings of 7th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Stanford, pp.96-101, Jun. 1998.