2000

An Object Model for Collaborative Systems and a Toolkit to Support Collaborative Activities

Haibin Zhu
New Jersey Institute of Technology, haibin@cis.njit.edu

Murray Turoff
New Jersey Institute of Technology, turoff@adm.njit.edu

Zheng Li
New Jersey Institute of Technology, zxl18078@njit.edu

An Object Model for Collaborative Systems and a Toolkit to Support Collaborative Activities

Haibin Zhu, haibin@cis.njit.edu [1,2]
Murray Turoff, turoff@adm.njit.edu [1]
Zheng Li, zxl8078@njit.edu [1]
[1]CIS Departent, New Jersey Institute of Technology (NJIT), University Heights, Newark, NJ 07102
[2] CSE Department, Changsha Institute of Technology (CIT), Changsha, Hunan, 410073, China

Abstract

The goal of a collaborative system is to provide a platform for group discussion so that the ideas of the majority can be captured and categorized. Such a platform would incorporate functionality to allow a group of experts to thoroughly explore and analyze a problem domain by following a discourse structure they could design, maintain and evolve as the knowledge structure for that particular domain. However, there are very few practical tools in current systems to support coordination strategy such as voting and scaling, and collaborative model building for learning. Any practical tool is better than an excellent theory. Our recent work is to design and implement a toolkit for collaboration. This toolkit supports the general tools for collaborative activities, and is easily accessed on the Web.

In this paper, we first illustrate the object model for collaborative systems; then, we discuss the basic requirements for collaborative systems that should be supported in the toolkit. The key problems of collaborative systems are also analyzed. Our proposal solution is to provide a collaborative toolkit. At last, we give the descriptions of this toolkit.

Introduction

The goal of a collaborative system is to provide a platform for group discussion so that the ideas of the majority can be captured and categorized. Such a platform would incorporate functionality to allow a group of experts to thoroughly explore and analyze a problem domain by following a discourse structure they could design, maintain and evolve as the knowledge structure for that particular domain (Turoff & Hiltz, 1999). However, there are very few practical tools in current systems to support coordination strategy such as voting and scaling, and collaborative model building for learning (Turoff, 1999).

Our concept of collaboration entails a situation where everyone is a potential equal contributor to a discussion transcript that becomes important as a memory for the group. Ideally such a transcript can evolve to become a knowledge base for the collaborators and those who use the results of the discussions. How to provide really useful software structures and tools to support large web communities on their knowledge work, i.e., assembling, making sense of, and working with very large collections of ideas and multi-media information? This is still to be clarified.

Our recent work is to design and implement a toolkit to support web-based collaboration. The toolkit could be embedded into any organization's current information systems, and could be easily accessed through the Internet. The toolkit provides various group decision-support tools and techniques such as voting, scaling, ranking, and provides useful processes like brainstorming, Delphi, etc. Our goal is to move useful group decision support techniques to the web to support dispersed large-scale collaborative activities and gain collective intelligence.

Object-oriented methodology is a general modeling methodology for information systems. We will take this model as our basic model for designing and implementing the collaborative toolkit.

OMCS: An Object Model for Collaborative Systems

Our work is based on object-oriented methodology, because it is accepted that object model can be used to describe and guide the design for almost any information system (Norman, 1996).

The Key Ideas for Object-Orientation (Zhu, 1992, 1998)

For object-orientation, we can insist on the following three principles:

- Everything in the world is an object;
- Any system is composed of objects (certainly a system is also an object);
- The evolution and development of a system are caused by the interactions among the objects inside or outside the system.

For a class of objects, we conceived that, Class ::= <Id, Ds, Ops, Intfc>, where

- Id is the identification or name of the class;
- Dd is the space description (template) for memory;
- Op is the set of operations the objects of this class can perform;
- Intfc is the unified interface of the objects of the class.

For the instances created from a class, also in the opinion of implementation, we conceived, Object ::= <Oid, Cid, Body>, where

- Oid is the identification or name of the object;
- Cid is the identification or name of the class of this object;
- Body is the actual space for this instance.

From this view, we can consider a collaborative system as an object that is an instance of a specific class. A well-defined class will become a practical model. We will discuss the OMCS model with defining elements of a collaborative system class.

The OMCS Model

In terms of object-oriented technology, we address that, any collaborative system can be an instance of the class CoSystemClass::=<CSID, CSDS,CSOP> (Zhu, Wang, Hu,1996, & Zhu, Wang, Shen, 1996), where

- CSID="CoSystem", it is an identification. A URL on the Internet can be taken as an identification.
- CSDS = <A, M, W, D> expresses the structure (or state) of the system, where
  - A denotes the platform architecture of the system;
  - M denotes the collaborative (coordination) mode;
  - W denotes the management model for information, W = Relation | MHEG| Dexter| DNCH| AMH| OCPN| Garg| Parumak|HAM|ODA|.....;
  - D denotes the support tools for discussing, D = Text|Audio|Video|Combined.
- CSOP=<R,S,E,I>, where
  - R expresses the services for roles, such as, chairman, lecturer, audience or others;
  - S expresses the services for cooperation, such as, start, enter, leave, late, leave early, group, ... ;
  - E expresses the services for editor, such as text editor, hypertext editor, multimedia editor;
  - I expresses the interface design for multi-user requirements, such as, WYSIWIS service, collaborative edit service.

So, if someone wants to design or implement a new collaborative system, s/he actually creates a new instance of this class, and set the instance body with new concrete attributes and new properties. We emphasize the structure A and the mode M for CSDS of the OMCS model.

The Structure of a Collaborative System

Concerning the basic structure for collaborative systems, we have three choices, i.e., A =Totally Centralized | Totally Distributed | Partly Centralized & Partly Distributed.

The totally centralized structure is a mainframe / terminal architecture. The collaborative system is put in the main frame, and the terminal just accepts the users' input and displays the information transmitted from the mainframe on the screen.

The properties of this type of structure are:

1. Easy to implement, but:
2. The input and output of the mainframe is the bottle neck of the system;
3. For N terminals, suppose the resolution is 640*480, every pixel needs 256 colors, and the I/O requirement is N*30*640*480*8*3=N*313.344 Mbps.

So, the conclusion is that this type of structure is not practical.

For the totally distributed structure, the system has peer-to-peer computers with the same function. The copies of the system reside on all the computers. It requires one copy for every user for the shared information. The problems with this architecture are:

1. Synchronization. Consistency among copies of shared information is difficult to control;
2. Needs high bandwidth network supports. The bandwidth equals N(N-1)(the interface operations + the operations for shared information), if there are N users for the system.

Partly centralized & partly distributed structure is evidently a client/server structure. There are many copies of the interface on different user clients (one for each), and there is only one copy of information management on the server. This architecture is a trade-off for the above two architectures, and has a relatively low bandwidths that is N (N-1) (the interface operations) + the operations for shared information, if there are N users for the system.

The Coordination Mode of a Collaborative System (Turoff, 1999)

There are four modes for general coordination activities, which are parallel, pooled, sequential and reciprocal. Any collaborative system must support one or more of the four modes for process control.

- In parallel mode, each individual approaches the problem entirely independently of the other members of the group. There may be information provided from each member to the group on the status of his or her progress, but there is no imposition of any group process upon the performance of the individual members.
- In pooled mode, the group imposes a structure or standard to capture and represent individual contributions into a collective group representation. There is no constraint on the sequence of activities that each member may undertake to arrive at individual results to be incorporated into the collective group results. A typical example is estimation of the budget in an organization.
- In sequential mode, the group imposes phases on the problem solving process that must be undertaken in a sequential manner by all the members of the group.

- In reciprocal mode, while individuals may independently work on the problem (as in the parallel mode), there is some form of consistency imposed upon the contributions made by the individual members. Furthermore there is a model agreed to by the group for the compilation and synchronization of the results. This is the dynamic process allowed by the incorporation of the consistency tools. It would never be possible without computer based group communications.

These four process control modes will be supported in our collaborative toolkit.

The General Services of a Collaborative System

For any object model, the services should be discussed after the attributes are introduced. The following services must be provided by any practical collaborative system.

Management of Collaborative Activities

Any system supporting collaboration should provide facilities like the following:

- How to inform:
  - Message earlier, Calling in time,...
- How to enter or leave in different time:
  - Late, leave early, ...
- How to designate the roles such as chairman, lecturer, audience, ...
- How to define tasks and priorities, because different roles in different tasks have different priorities at different times.
- How to add and control comments, such as comments link, and modification records.

Group-Aware mechanism

This is the most important mechanism for a synchronized collaborative system. It means that one user must be aware of others' existences. The mechanism is categorized in two dimensions:

- Synchronized: also called view sharing (WYSIWIS): Every user must see what others see.
- Asynchronized: a user could be aware of others from the state of the shared information.

The mechanism can also be divided into the following two categories:

- Explicitly Aware: This is the main goal for collaborative systems which make people understand each other more easily
- Implicitly Aware: The collaborative systems should make use of the implicit aware information.

Asynchronized cooperation will give the collaborators the ease to participate in the collaborative activities any time from any where. But it will hinder the explicit awareness. The system must make some efforts to make as much explicit awareness as possible by comments, notifications etc.

A Toolkit for Collaboration

The functions discussed above will be incorporated into our collaborative as the basic features. The toolkit uses the OMCS model discussed above.

For a web-based environment, the collaborative toolkit would support:

- Defining and creating a problem/topic.
- Forming alternatives on the problem/topic.
- Gathering and manipulating a list of criteria concerning the specific problems.
- Scaling such as Likert scale, semantic differential, "Thurstone's Law" applied to lists.
- Voting and weighing schemes applied to single criteria items or lists of items.
- Being able to collect lists from an outline word file, as well as being able to collect data from a form routine on the user's machine as to voting on items or on lists.

The toolkit will have such functions:

- Allow users to collect problems for decision-making, set up and manage routines for collecting criteria items concerning the problems, including establishing the problem-solving group, nominating member roles, gathering criteria list, edit criteria list etc.
- Under network environment, users can access the voting system freely at any time, any place through web-browser and voting and weighing on the criteria items. Users have to authenticate themselves (not included in this sub-system). They can also modify their voting results at any time freely.
- Voting results will be processed dynamically under certain conditions using "Thurstone's Law". Users can check the voting results on-line freely.

The Underlined Tools in the Toolkit

In the toolkit, the following tools would be provided:

Tool$_1$: Secretary tool. Collect the opinions from individuals, and form a combined candid opinion without any prejudices for the group. This tool should be better than the human secretary, since a human secretary might have bias towards some specific topics. This tool should incorporate some basic cognitive or psychology laws to make it more practical).

Tool$_2$: Voting and scaling tool. Any combined opinion should be voted and scaled by the group members. The basic methods are simple majority, weighted majority, and weighted sum.

Tool$_3$: Communication tool. Any individual could exchange opinions with any member. We can apply some products such as E-mail, private conference or other tools for exchanging opinions.

Tool$_4$: Retrieval (filtering) tool. Automatic information retrieval is needed for a group or for an individual.

Tool$_5$: Role tool. Manage the roles for different members of a group.

Tool$_6$: Evaluation tool. Evaluate the user's opinions and change his or her weight for voting and other activities.

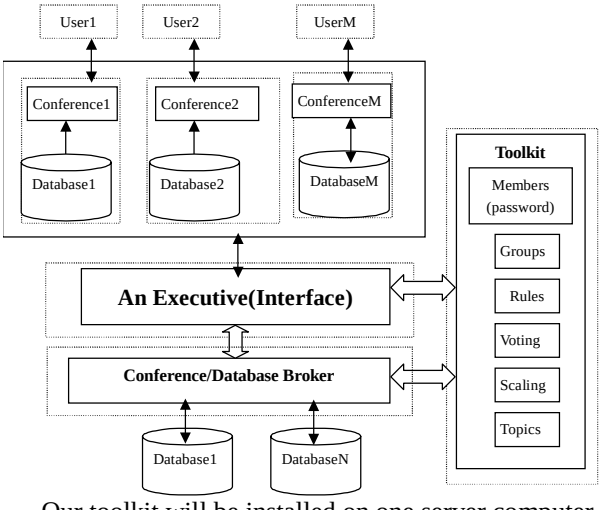
Tool$_7$: Group-awareness tool. Help to be aware of each other, such as comments or suggestions for a topic.

We can also add new tools continuously to meet new requirements.

The Architecture of the Collaborative Toolkit

With the OMCS model, we can consider the toolkit as an object and the tools as the services of the object. Any other objects such as conferencing systems or web browsers can interact with this object to get services. Therefore, we consider our toolkit as a partner for a collaborative system such as a conferencing system of Virtual Classroom. The logical architecture for the toolkit corresponding to the conferencing systems is shown in Fig.1.

Figure 1. The Logical Architecture for the Application

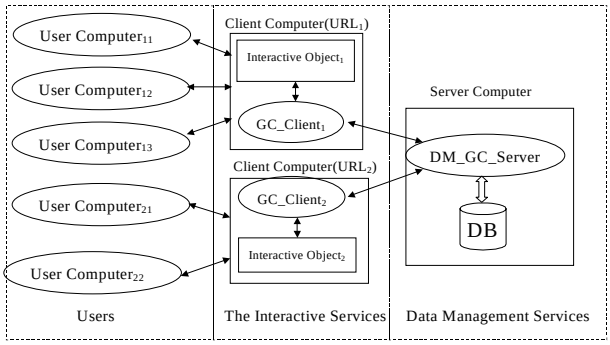


Our toolkit will be installed on one server computer and many client computers. The users can use the toolkit for collaborative activities by the URLs of the client computers. After the server software is installed, you can install any other client software in any client computer. Therefore, the implementation architecture is a three-tier architecture. That is to say, in the opinion of the model OMCS, the architecture A of the toolkit is a two-tier Client/Server architecture, or partly centralized & partly distributed (Fig.2).

The users only know that the toolkit is on the client computer denoted by a URL. In the client computer, there is an interactive tool for the user's interactions with the toolkit. The data management tool and the group communication tool are transparent to the ordinary users.

In the client computer, only a small database is needed to store the temporary information of groups, members and topics. All the permanent information will reside on the server computer. The interactive tool is mainly composed of two objects that are the interactive object and the group communication client object.

Figure 2. The Implementation Architecture



The server will support all the services for managing all the data or information of all the members, groups and topics relevant to any kinds of collaborations. There may be many client computers installed with the interactive tool. The interactive tool is the server for the user computers. On a user computer, a user might open a web browser to interact with the interactive object.

So, there are three major objects in the toolkit, the Interactive_Object serving user interactions, the GC_Object serving group communications and DM_GC_Server serving data management and group communications.

Why did we construct this architecture? There are many advantages for technology requirements.

First, this architecture will support modularity. There are only three major modules for this architecture. As a side effort, we may provide a set of tools for group communication services. Second, this architecture supports the sharing of the collaborative information. Other groups may share the decisions made by one group. Third, the messages can be sent immediately to the group members with the group communication tools. Also, with the group communication tools, a member of a group can send messages to individuals, to groups or to the public. Fourth, the data on the web are decreased, because the users interact with the client computer and no communications among the user computers are needed. Fifth, you can install many copies of the interactive object to decrease the data transferring on the web, and the user computers can communicate with the nearest client machines. Last, if reliability is required in a high priority, we can add another duplicate server for the management of the shared information.

As a whole, we can use the following formula to describe the toolkit.

The toolkit =<"NJIT Co-Toolkit", ToolkitDS, ToolkitOP>, where

- ToolkitDS = <Two-tier Client/Server architecture or partly centralized & partly distributed, Parallel | Pooled | Sequential | Reciprocal |, Relational Database Model, Text>;
- ToolkitOP = <Moderator | Member | Listener, Start |Join | Leave | Leave early | Vote | Scale | Search, Text-editor, Web-style user interface>.

In other words, the toolkit supports text-based collaboration. The information management model is the relational model, and the architecture is based on client/server. It will support different modes of coordination, such as parallel, pooled, sequential and reciprocal modes. It will also provide different services for collaborative activities in accordance with different role control. The user interface is web style-based text-editor.

Summary

From above, we discussed an object model for collaborative systems. This model can be taken as a template to build or evaluate a collaborative system. Also, we discussed the toolkit we are developing based on the current requirements for collaborations on IT fields. With the architecture of the toolkit, it would be very easy for adding new tools into it. We are now doing the design for the interactive tools and the design for data management tools, the group communication tool will be coded soon. We hope we can accomplish our toolkit soon and make it available to the IT society.

Acknowledgments:

This project is supported by NSF, CISE-ITO, “Studies of Distributed Multimedia Support for Group Collaboration via the Web.” Thanks to all the participants of IS seminar of CIS department. Also thanks to the funding of CSC (China Scholarship Council).

References

Turoff, Murray, An End to Student Segregation: No More Separation Between Distance Learning and Regular Courses. *A summary of the invited plenary for the Telelearning 99 meeting in Montreal, Canada, November,* 1999.

Hiltz, S. R. and Wellman, B. Asynchronous Learning Networks as Virtual Communities. *Communications of the ACM, Sept.* 1997.

Norman, Ronald J., Object-Oriented System Analysis and Design, Prentice Hall, 1996.

Rana, A. R., Turoff, M., and Hiltz S. R., Task and Technology Interaction (TTI): A Theory of Technological Support for Group Tasks, *Proceedings HICSS,* 1997.

Turoff, M., Virtuality, *invited paper for special section of CACM, Volume 40,* Number 9, September 1997, pp. 38-43.

Turoff, M. & Hiltz, S. R., Supperconnectivity, CACM, July 1998.

Turoff, M., J. Fjermestad, A. Rana, M. Bieber, R. Hiltz, Collaborative Hypermedia in Virtual Reality Systems, *Proceedings of the third Americas Conference on Information Systems,* August, 1997.

Turoff, M., & S. R. Hiltz, Software Design and the Future of the Virtual Classroom, *Journal of Information Technology for Teacher Education, Volume 4*, Number 2, 1995, 197-215.

Turoff, M., S. R. Hiltz, A. N. F. Bahgat, and A. Rana. Distributed Group Support Systems, MIS Quarterly; December 1993, 399-417.

Turoff, M., Rao, Usha, and Hiltz, Starr R.. Collaborative Hypertext in Computer Mediated Communications, *HICSS 91*, January.

Zhu, Haibin, Wang, Pu, and Hu, Shouren, "CoWNCH: An Object-Oriented Model for Computer Supported Hypermedia Co-Authoring", In *Proceedings of 1995 European Simulation Symposium (ESS'95)*, Erlangen-Nuremberg, Germany, Oct., 1995.

Zhu, Haibin, Wang, Pu, and Hu, Shouren,"The Design of a Multimedia Co-authoring Platform based on Client/Server Model", *ACTA Electronica Sinica*, Vol 25(5), 1997.

Zhu, Haibin, Wang, Pu, and Hu, Shouren, "A Multimedia Co-authoring System Object Model AMWD/RSEI", *Chinese J. of Computer, Vol 20(Supplement)*, 1997.

Zhu, Haibin, Wang, Pu, and Hu, Shouren, "An Object-Oriented Multimedia Management Model WNCH", *Software Journal, Vol 7(Suppl.)*, 1996.

Zhu, Haibin, Yang, Guogui, and Liu, Zhong. Object Oriented Principle and its Applications, Publishing House of Changsha Institute of Technology (CIT), Sept. 1998.

Zhu, Haibin, Cai, Kaiyu, Fan, Aihua and Song, Hui. Principles and Designs of Distributed Systems. Publishing House of CIT, Sept. 1997.

Zhu, Haibin. Object Oriented Technology: Principles and Designs, Publishing House of CIT, Oct. 1992.