*Innovations in Education and Teaching International,*
*Vol. 42, No. 4, November 2005, pp. 349–362*

Routledge
Taylor & Francis Group

# Supporting real-time collaborative learning with web-based groupware

Stephen C. F. Chan*, Cane W. K. Leung, Cassidy Y. Yeung, Teddy C. Y. Chow, Edward W. C. Tsui and Vincent T. Y. Ng
*The Hong Kong Polytechnic University, Hong Kong*

The Wireless Online Tutoring System (WOTS) is a web-based, synchronous groupware system that has been developed to facilitate student collaboration in group projects. It supports interactive and collaborative structured-diagramming and is equipped with group awareness features such as remote cursors, locking, and text-based instant messaging. Users can access WOTS from both personal computers and handheld devices, allowing it to be used in wireless classrooms or distributed environments. Tests on student software design projects validate the ability of WOTS to support student collaboration in group projects, and show that its architecture is scalable and reliable. Its performance is expected to improve as more powerful handheld devices and wireless technologies come on to the market.

## Introduction

Collaborative learning is a philosophy of teaching in which learners work together on a common goal, exchange their opinions on a subject, clarify the meanings of concepts or jointly address a problem (Hron & Fredrich, 2003). A more detailed definition was given by Landsberger (2003): members participating in a team develop and share a common goal, contribute their understanding of the problem, respond to and work to understand others' questions, insights and solutions, and are accountable to and interdependent on each other.

Collaborative learning has a number of advantages. Studies have shown that learning is enhanced when it is more like a team effort than a solo race (Gerdy, 1998, as cited in Wiersema, 2000), and that, compared to learning alone, learning in small groups improves critical thinking (Gokhale, 1995). Further, students who are encouraged to contribute and participate learn how to learn and are less dependent upon a teacher (Wolz *et al.*, 1997). Collaborative learning, however, is not unproblematic. In one common form of collaborative learning, group projects, students may find it difficult to schedule face-to-face meetings; or may perceive the distribution of work as being unfair. With the advent of the World Wide Web, such problems can be obviated

*Corresponding author. Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Hong Kong SAR, China. Email: csschan@comp.polyu.edu.hk

ISSN 1470–3297 (print)/ISSN 1470–3300 (online)/05/040349–14

DOI: 10.1080/14703290500268469

and more effective support can be provided for collaborative learning through the use of web-based tools and wireless handheld devices. In this paper, we introduce our work on the Wireless Online Tutoring System (WOTS), a web-based tool designed to support the collaborative work of students.

Collaborative learning through group projects is a commonly-used approach in many universities, including The Hong Kong Polytechnic University's Department of Computing. For example, in an undergraduate course on object-oriented technologies (a method for developing information systems), students were assigned a group project to design and develop an online trading platform. A popular software development methodology, Unified Software Development Process (Booch *et al.*, 1999), was discussed in a series of lectures in parallel with the execution of the project. Such a project can be executed in three stages:

1. *Requirements Specification*. Students were given a project description. Based on the description, they were required to submit a report on the business model of the e-trading platform and to specify the requirements of the project.
2. *Systems Analysis and Design*. At this stage, students were required to develop system design diagrams in Unified Modelling Language (UML) (Booch *et al.*, 1999), a graphical modelling language commonly used in information system design based on the object-oriented approach (Martin & Odell, 1992).
3. *Implementation and Testing*. Students were required to implement their designed system using specified techniques, and to then submit a final report containing revised versions of the deliverables of the previous stages, as well as additional sections related to the implementation and testing results.

Most commercial educational software available in the market, such at WebCT (http://www.webct.com), and research software, such as CECIL (Gardner *et al.*, 2002), focus on supporting flexible learning and course management. It is rare to find tools for a specific task that directly support the collaborative work of students. For example, most currently available UML editors are designed for a single-user environment, yet system analysis and design processes usually involve a group of people. It is for this reason, to support students' collaboration in executing software development projects, that WOTS has been developed.

In general, collaboration can take place in two modes, thus collaborative groupware can also be divided into two categories. The first category is synchronous groupware, where participants meet at the same time. The second category is asynchronous groupware, where participants connect at different times and leave their work or messages for others to read and comment on. A study that compared the difference between synchronous and asynchronous groupware has shown that synchronous collaboration led to more dynamic interactions, more quickly resolved issues and shorter work periods (Walker *et al.*, 2000). In view of these advantages, WOTS was designed to support synchronous collaboration.

## A group project in teaching information system design and development

In the Introduction, we described the group project used in the teaching of a course on information system design and development based on object-oriented technologies. In this section, we provide details of the project, followed by the requirements for WOTS that were derived from the project.

The project was to design and develop an e-trading platform for building and construction companies. Students were grouped by mutual consent into teams of at most four. These teams were required to develop UML analysis and design models, and then implement them. The e-trading platform had two major functions: to provide trading information and to facilitate trading transactions. Users would log into the system as either sellers or buyers of construction materials. Sellers could upload their product information and handle buyers' orders. Buyers could search for certain types of products or they could browse the products on display. When the products requested by a buyer were found, information on the product would be displayed together with the seller's information. The buyer could negotiate with the seller about the price or, by filling in online forms, place an order.

As described in the Introduction, execution of the whole project can be divided into three stages, Requirements Specification, Systems Analysis and Design, and Implementation and Testing. The development of UML modelling is found in the second stage and can be broken down into a further three phases:

1. *Draft the Preliminary Solution*. One or more members propose the preliminary UML models through collaboration. The models may include major objects and their relationships in the systems.
2. *Refine the Solution*. This stage could produce different types of discussion.
   (a) *Discussions about UML concepts*. Depending on their proficiency in UML modelling, some members may understand better (or worse) about certain UML concepts than others.
   (b) *Discussions about the developed UML models*. Based on the preliminary models, everybody contributes what they think is right to the diagram. Most of the discussions and interactive activities occur at this phase.
3. *Conclude the Project/Solution*. After several discussions, a conclusion can be drawn and a solution agreed upon. The length of the discussions depends on the proficiency of group members on the topic and complexity of the given problems.

### *Requirements of WOTS*

WOTS was designed to support student group projects that were using UML modelling. Given the increasing popularity of mobile learning in wireless and *ad hoc* classrooms (Chang *et al.*, 2003), it was also regarded as desirable that it should be adaptable to wireless handheld devices. Based on these objectives, several requirements of WOTS were identified.

First, as students work together as a group throughout the whole modelling process, concurrency control should be implemented in the UML editor. Concurrency control is concerned with coordinating the work of users and with preventing conflicting operations, which may arise when more than one user attempts to manipulate the same UML object in a UML diagram.

Second, a text-based chat-room should be provided to facilitate different types of discussion among students. A chat-room is needed because it may be difficult, and probably inappropriate, to represent discussion topics such as project requirements and UML concepts on the UML editor.

Third, an archiving feature should be provided for any UML diagrams that are produced. To allow easy reuse and retrieval for submission or further discussions, archived diagrams should have a standardized format.

Fourth, apart from concurrency control, it would be worthwhile to consider other essential groupware design issues. In our context, such issues include group management and provision of group awareness. Group management is required to manage the students' access to the system and help them create or find out an available group to join. Group awareness is believed to be significant in supporting effective collaboration, as participants in a group can perceive each other's presence.

Fifth, the system's robustness in terms of server performance should be ensured. System responsiveness and reliability are significant issues in a groupware system (Brinck, 1998). There should be measures to improve system scalability and to handle server failure.

Finally, the design of the system should take into consideration the constrained environment of handheld devices. Because of its popularity, we chose a Pocket PC as the handheld device for the development and evaluation of WOTS. It has the typical hardware characteristics of handheld devices that WOTS should cater for: a small screen size, low processor speed and stylus-based input.

## Related work

WebCT is a popular and comprehensive web-based course management tool, but its functions are generic. For example, its whiteboard provides paint functions but no structured diagrams. NetCha*t* (Kreutz *et al.*, 2000) is a web-based communication and collaboration tool designed for educational purposes. It provides a chat-room, newsgroup and graphical whiteboard features, but its whiteboard is very simple and cannot support complex conceptual diagrams. ArgoUML (http://argouml.tigris.org) is a Java-based open-source UML tool developed for a single-user environment. It supports many kinds of UML diagram and allows easy interchange of diagrams with other UML editors such as Rational Rose. The system cannot store or retrieve graphical information such as the layout of the saved UML model. VortexX (Ratcliffe *et al.*, 2003) is a tool that supports collaborative work with UML which is more focused on documenting the development process.

WOTS has four significant advantages over these UML tools and web-based collaborative systems:

1. It supports real-time collaborative structured diagramming (UML modelling). Once diagrams, including UML model information and graphical layouts, are drawn in WOTS, they can be archived and retrieved using standard formats.
2. It supports instant messaging for group discussion (chat-rooms).
3. It is accessible through desktop computers as well as through handheld mobile devices.
4. It is scalable, distributed and reliable.

## The WOTS architecture

WOTS is a distributed, scalable and reliable system that provides UML modelling and text-based chat support. The front-ends of WOTS are web-based. They can be accessed through

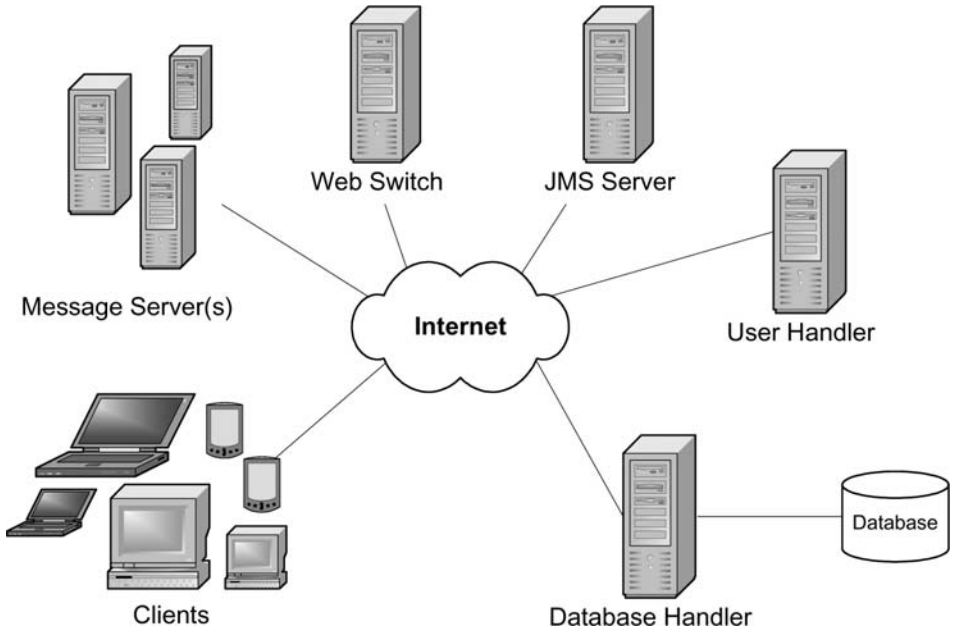


Figure 1. Architecture of WOTS

personal computers or handheld devices with Internet connections, allowing mobile learning in *ad hoc* or mobile classrooms.

As can be seen in Figure 1, the WOTS architecture consists of clients and five key server components (Chow *et al.*, 2002): Message Server, Web Switch, User Handler, Database Handler, and JMS Server. The JMS Server is a message-oriented middleware (MOM) facility (Monson-Haefel & Chappell, 2001) that allows communication between servers. It is capable of handling hundreds of messages every second.

When a student logs into the system, a request is sent to the Web Switch, which is used to keep track of the state of each Message Server, such as its workload or server failure, and will connect the student to the Message Server having the smallest workload.

The student can then create a group or select a group to join. The Message Server serving the student will inform the User Handler to keep track of the user's login/logout state. Other students in the same group will also be informed of the user's login event.

After a student has joined a group, he/she will receive messages being passed in the group, or convey messages to other collaborating students. All these messages are handled by the Message Server. They will be archived in the system's database by the Database Handler. The student will also be able to view and edit the UML diagram being developed.

When a student logs out, the User Handler will update that student's state. Students in the same group will be informed of that student's logout event.

In the above workflow, the servers in WOTS are required to communicate with each other through the JMS Server. Since they are connected to the JMS Server using their Internet Protocol (IP) addresses, they can be distributed among different computers at different places.

In addition to being distributed, the architecture of WOTS has a number of features that ensure its robustness and which allow it to be easily deployed in the academic environment at a low cost. WOTS is Java-based and web-based, making it accessible to all Java-enabled standard web browsers. It is also scalable and remains responsive when a large number of students are connected to the system. Load balancing and fault tolerance mechanisms have also been implemented to improve system reliability.

## System design

This section describes the student-accessible functionalities of WOTS in three main categories of functionality: those that facilitate group management, those that support text-based group discussions, and those that allow synchronous, collaborative UML manipulation. This section also explains the design rationales of the functions with reference to the requirements identified in the section 'Requirements of WOTS'.

### *Group management*

Group management encompasses a number of issues:

- Deciding who is qualified to join the system, usually by predefined credentials.
- Identifying the groups that are available.
- Enabling students to identify their partners.
- Allowing students to leave a group at any time.

For these purposes, WOTS maintains information about those students who have access rights to the system, the currently available groups, the students who have logged into the system, and the group they joined. When students want to access WOTS, they must first provide their credentials. After authentication, they can decide to create a group or to join an existing group and start working on their project.

### *Instant messaging for group discussion*

Instant Messaging (IM) is widely used and studied in distributed communication and collaboration such as in Handel and Herbsleb (2002) and Hansen and Damm (2002). It is essential in WOTS because when students work together on a group project, different kinds of discussion may take place. Topics such as project requirements, UML concepts, or preliminary solutions may be difficult to represent using the UML editor. For such messages, WOTS provides students with a text-based chat-room.

### *Collaborative UML manipulation*

*Coordinating synchronous collaboration.* Apart from the basic UML editing operations, several features have been introduced in WOTS to address the design issues in supporting the synchronous collaboration of students. Such design issues include concurrency control and provision of group awareness.

WOTS adopts lock-based concurrency control of shared UML objects (Chow *et al.*, 2002). For example, when a student wants to edit a particular UML class, he/she must first acquire a lock on it. This means that no two users can edit the same UML object simultaneously. Nonetheless, synchronous collaboration is still possible as different users can be contributing to different parts of the diagram at the same time. WOTS employs a colour scheme to indicate which UML objects have been locked, and by whom.

Group awareness ensures effective collaboration as the participants in a group can perceive each other's presence. It can be provided by the following mechanisms (Ceglar & Calder, 2001):

- Status, the information regarding the users in collaboration.
- Events, the means by which real-time workspace awareness is maintained.
- Filtering of awareness information, required for privacy and to reduce information flow.

WOTS addresses these mechanisms using the remote cursor, lock requests, and text-based chat-room features (Chan *et al.*, 2003). In WOTS, each student can activate or inactivate his/her remote cursor, which can draw other people's awareness to a certain piece of information by 'pointing' at it when activated.

A lock request can be used when a student wants to edit an object that is locked by someone else. For example, if student A wants to edit an object that is locked by student B, A may wait until the lock is released or send a message to B, informing B that he/she is waiting for the object to be unlocked.

The chat-room facility of WOTS provides group awareness because, in addition to displaying messages originating from the users in a group, it also shows the group's login and logout events and notifies users about special events in the UML editor, such as lock requests.

*Visualizing and navigating UML diagrams.* One major difference between a desktop computer and a Pocket PC is the size of their screens. The screen size of a Pocket PC is so small that it can display just a few UML objects at a time. A number of features were therefore introduced in WOTS to explicitly address difficulties in visualizing and navigating UML diagrams. They include zooming and searching UML diagrams, and tracing relations among UML objects.

The zooming options provided in WOTS include Zoom In, Zoom Out, Show All Objects, and Restore to Original Size. The level of detail being displayed will vary automatically according to the zoom rate of the current view. Figure 2 shows the student's view of the Pocket PC front-end.

The search function allows students to input a keyword to be searched, and this returns a list of objects containing that keyword. After performing a search, the viewport, the visible portion of the drawing canvas, will shift to the location of the first object in the search results. Show Next and Show Previous options allow the next and previous objects in the list of search results to be displayed.

The trace feature was designed to help students discover more directly the relationship among various UML objects. Instead of looking for keywords, this feature considers the relationships among objects. A trace can be done on different types of relationship: directly related objects, all related objects, superclasses, and subclasses. After performing a trace, the optimal zoom rate for displaying all the objects found in the trace is computed and the display then refreshed.

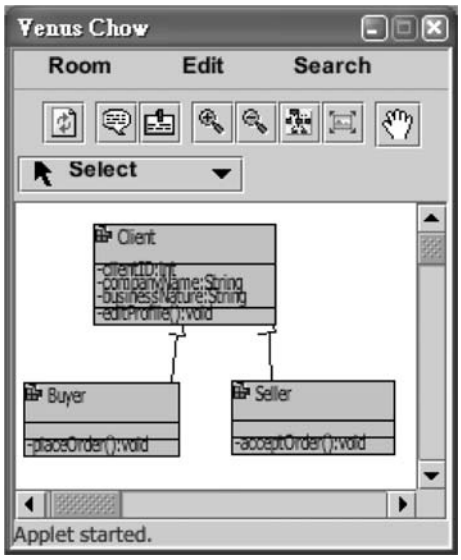


Figure 2. UML diagram on a handheld device

*Archiving UML diagrams produced in collaboration.* Students using WOTS to execute their projects were expected to produce system designs in the form of UML diagrams. An archiving mechanism was therefore developed to help students store their developed models for submission or further discussion. Archived diagrams can be viewed in standard web browsers.

## PDA adaptation

Since the screen size, system environments, and hardware power of Pocket PCs are very different from those of personal computers, the client front-end is in two different versions. In general, the Pocket PC front-end is a restricted version of the personal computer front-end.

### *Pocket PC as a development platform*

The development tools and software packages that are available for Pocket PCs are not only limited, they may vary according to device types. In order to achieve portability between device types or even operating systems, Java, which has a cross-platform capability, was chosen as the development language. WOTS was developed as a Java applet, so that students would not need to pre-install WOTS on their devices. They can access the system using a Java-enabled web browser. This is desirable in a mobile or *ad hoc* classroom environment.

### *The Pocket PC front-end*

Due to the limited screen size of a Pocket PC, the Pocket PC front-end has to be designed differently from the personal computer front-end. The workplace in the personal computer front-end

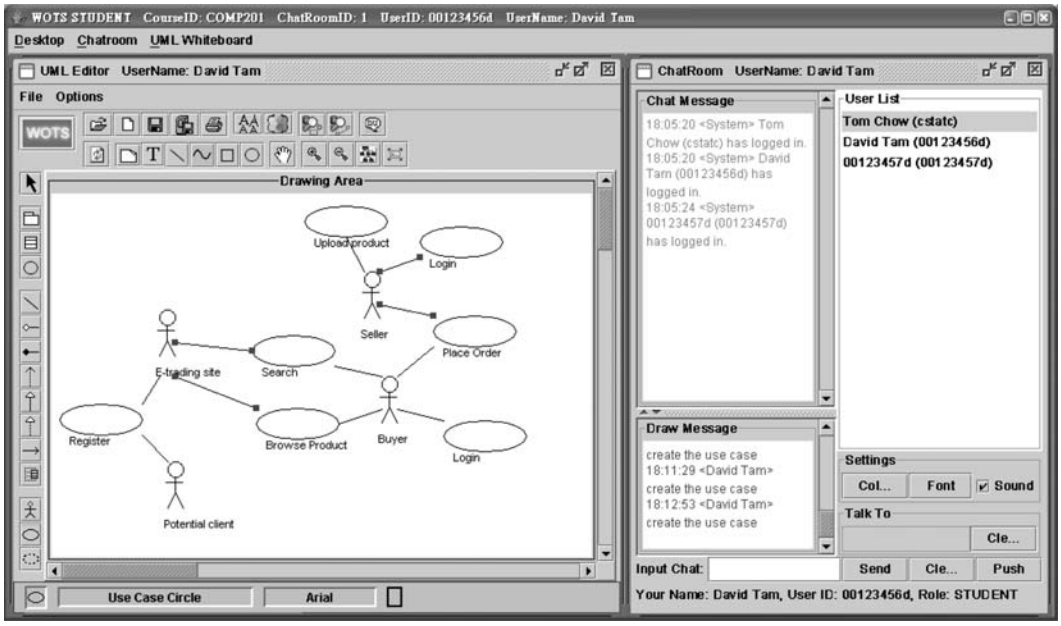


Figure 3. Preliminary use case model developed in Phase 1

displays several components. As shown in Figure 3, these include the UML editor, a chat window, a message window describing the operations performed on the UML editor, and the list of participants in the group. The Pocket PC front-end, however, cannot display all of these components on-screen. Instead, after the user has logged into the system, only the UML editor is shown. Students can switch to the chat window and back again by using a button and a menu option. Invoking the 'User List' option from the menu can be used to display the list of students in the group.

## Application and validation

WOTS was evaluated according to two main criteria. The first criterion was its server performance and reliability. The second criterion was its ability to support student collaboration in software development projects using UML modelling.

### *Support for collaborative learning*

The project used in the evaluation was representative of collaborative learning activities. First, students were genuinely working together towards a common goal. Second, as they discussed UML concepts and the models they were developing, students were involved in clarifying the meanings of concepts and exchanging opinions on related issues. Finally, when, at the conclusion of their projects, students exported their developed models, we could see that their collaboration had produced successful outputs.

*Test cases in software development project*

As aforementioned, collaborative development of UML models consists of three phases. They are:

1. Draft the preliminary solution.
2. Refine the solution.
3. Conclude the project/solution.

Three test cases were therefore designed to access the ability of WOTS to support the activities involved in these phases. In the following test cases, three students, A, B and C, are collaborating in the same group. A and B are accessing WOTS on their own desktop computers at home while C is using a Pocket PC somewhere in the school's library. After reviewing the project requirements, they want to develop a use case diagram for the system. Each of the following test cases includes a description of the stage executed, the WOTS functions used in the test case, a sample of the students' dialogue, and brief results of the test, which are further discussed in the next section.

*Test case 1:*

| | |
|---|---|
| Phase executed: | (1) Draft the preliminary solution |
| Supporting functions in WOTS: | Chat-room discussion, locking/unlocking of objects, request lock feature, UML editing feature |
| Sample situation: | A: It will be easier for us to start by identifying the actors first.<br>B: Actors should include Buyer and Seller<br>*[B adds these actors to the diagram]*<br>C: And also potential client and E-trading site<br>*[C adds these actors to the diagram]*<br>A: Should I draft the associated use cases first, and then discuss them together?<br>B: Fine.<br>C: O.K.<br>*[B and C possess the locks of all actors, thus A sends them lock requests]*<br>*[B receives the message and releases the lock]*<br>*[A locks the Buyer actor and starts modelling]* |
| Test result: | Phase 1 accomplished with satisfactory performance. A preliminary use case model was drawn (Figure 3). |

*Test Case 2:*

| | |
|---|---|
| Phase executed: | (2) Refine the solution |
| Supporting functions in WOTS: | Chat-room discussion, locking/unlocking of objects, request lock feature, UML editing feature, remote cursor, zooming, searching |
| Sample situation: | *[B and C read the use cases modelled by A]*<br>*[B found that both the Buyer and Seller actors have a use case called 'Login']*<br>B: How come there are two use cases called 'Login'?<br>*[C is using a Pocket PC and cannot see the use cases. He zooms out view and performs a search on the keyword 'Login']*<br>C: We made a mistake here. There should be another actor, say 'Client'. 'Login' should be a use case of 'Client'<br>*[C sends lock requests for the related objects]*<br>*[A receives the messages and releases the locks]* |

| | |
|---|---|
| | *[C adds the actors 'Client' and performs a trace on 'Buyer' and 'Seller' to find the two 'Login' use cases. He then removes one of the 'Login' use cases, and relates the other one with 'Client']*<br>*[A mistakenly adds association lines to relate the actors 'Client', 'Buyer' and 'Seller']*<br>B: 'Client' should be a 'Generalization' of 'Buyer' and 'Seller'!<br>*[A corrects the mistake]* |
| Test result: | Phase 2 accomplished with satisfactory performance. A refined use case model was drawn (Figure 4). |

*Test Case 3:*

| | |
|---|---|
| Phase executed: | (3) Conclude the project/solution |
| Supporting functions in WOTS: | Chat-room discussion, save feature |
| Sample situation: | A: Do you all agree that our solution is good enough?<br>B: Yes.<br>C: So we have to save this diagram for our final report.<br>*[A invokes the 'Save' function]*<br>*[Two files are saved, one in XMI format and the other in SVG format]* AQ1<br>*[The use case model is completed and the discussion ends]* |
| Test result: | Phase 3 accomplished with very satisfactory performance. The developed model was saved for future use. |

*Server performance of WOTS*

The server performance of WOTS was evaluated by simulation. Tests performed include adding and shutting down Message Servers while the system was operating, and heavily loading the

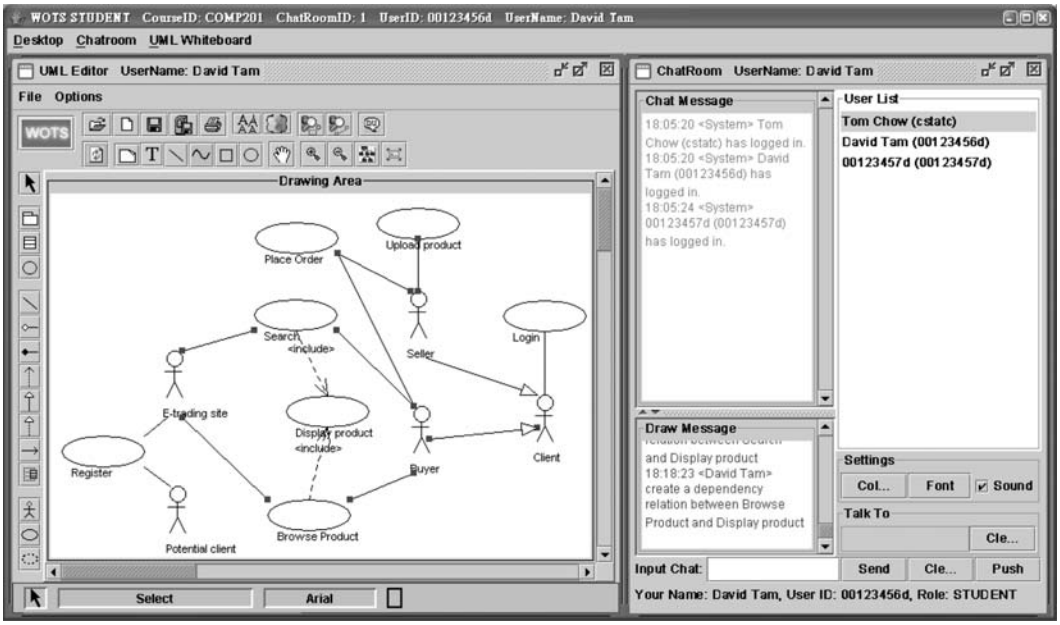


Figure 4. Refined use case model developed in Phase 2

system with hundreds of clients. WOTS performed as expected: without deadlock, exceptions, or runtime errors.

The performance of WOTS was further measured under various loading conditions. As expected, it was found that both the mean process time per user request and CPU utilization decreased as more Message Servers were added. However, the experimental results also showed large random variations when client request turnaround-time was measured from the client side. It appears that the network performance, affected by unpredictable factors, dominates the server processing time.

### Discussion of test results

The previous section described several test cases derived from the execution of a software development project. All of the test cases were accomplished successfully, but there was variation in the degrees of satisfaction with which they were accomplished. Although the Pocket PC front-end has been tested to be functional, its performance is only fair. This is chiefly a product of the constrained environment, both hardware and software, of Pocket PCs.

The screen size of a Pocket PC is too limited. Although the visualization and navigation functions provided do assist users in viewing and editing UML models, many extra mouse clicks were required for zooming and scrolling. Java support on Pocket PCs is also inadequate. In order to execute WOTS on a Pocket PC, the installation of a Java Virtual Machine (JVM) would be required. When executing Java applications and Applets, however, serious delays occur, and this directly degrades the performance of the WOTS Pocket PC front-end.

Table 1 summarizes the test cases and their results in the aforementioned software development project. Fulfilment scores were based on users' satisfaction and on an evaluation of the effectiveness and efficiency of their work. The range of the fulfilment score is 1 to 5. The higher the fulfilment score, the more satisfactory was the performance of WOTS.

Table 1. Summary of test cases in software development project

| Phase | Description of test case | Supporting functions in WOTS | Fulfilment score |
|---|---|---|---|
| 1 | Draft the preliminary solution | UML editing functions include adding, moving, modifying, and deleting UML elements and relations, and exchanging messages with co-users in chat-room. | 4 |
| 2 | Refine the preliminary solution | UML editing functions include adding, moving, modifying, deleting UML elements and relations, using visualization and navigation options (zoom, search and trace) to review UML models, exchanging messages with co-users in chat-room, and enhancing collaboration using different group awareness functions (remote cursor, locking, lock requests). | 3.5 |
| 3 | Conclude the project/solution | Exchanging messages with co-users in chat-room to conclude project/solution, and save the diagram in XMI and SVG formats for future retrieval. | 5 |

## Conclusion

This paper has outlined the features, advantages and some of the pitfalls of collaborative learning and has introduced a web-based collaborative tool, WOTS, a synchronous groupware system developed to support students' collaboration in group projects, a kind of collaborative learning. WOTS was designed in response to some of the shortcomings of most currently available educational software in the research area. Such software seeks to support flexible learning and project management, but provides little support for student collaboration, particularly in synchronous collaboration.

WOTS facilitates students' collaboration in group projects through a UML editor and a text-based chat-room. It can be accessed using a desktop computer or a Pocket PC in a wired or wireless environment. Tests have validated its ability to support students' group collaborative work, and have shown that it has a scalable and reliable back-end architecture. Although the performance of the Pocket PC front-end of WOTS is only fair, this is chiefly the result of the hardware and software constraints of the Pocket PC platform. It is expected that the performance of WOTS will improve in the future as more powerful and mature wireless devices and technologies come on to the market. The current version of WOTS supports a specific form of structured diagram designed for the development of information systems, but the features are generic enough that it should be fairly straightforward to extend it to support other disciplines where structured diagrams are used extensively.

## Acknowledgements

The work reported in this paper was partially supported by Hong Kong Polytechnic University Research Grant H-ZJ82, and Learning and Teaching Development Grant LTG98-99/COMP014.

## Notes on contributors

Dr Stephen Chan is an Associate Professor and Associate Head of the Department of Computing at The Hong Kong Polytechnic University. He is currently working on the development of collaborative web-based information systems, with applications in education, electronic commerce and manufacturing.

Cane Leung is a research student in the Department of Computing at the Hong Kong Polytechnic University. Her research interests are computer-supported collaborative work and collaborative filtering.

Cassidy Yeung is currently a masters degree student at the University of Southern California.

Teddy Chow is a research student in the Department of Computing at the Hong Kong Polytechnic University. His research interests are distributed and mobile computing.

Edward Tsui is a research student in the Department of Computing at the Hong Kong Polytechnic University. His research interests are computer-supported collaborative work.

Dr Vincent Ng is an Associate Professor in the Department of Computing, The Hong Kong Polytechnic University. His research interests include databases, data mining, XML and Internet computing.

## References


Bernstein, P. A., Hadzilacos, V. & Goodman, N. (1987) *Concurrency control and recovery in data systems* (Reading, MA, Addison-Wesley).

Booch, G., Rumbauch, J. & Jacobson, I. (1999) *The Unified Modelling Language user guide* (Reading, MA, Addison-Wesley).

Bourke, T. (2001) *Server load balancing* (Sebastopol, CA, O'Reilly).

Brinck, T. (1998) *Groupware: introduction.* Available online at: http://usabilityfirst.com/groupware/intro.txl (accessed 20 October 2003).

Ceglar, A. & Calder, P. (2001) A new approach to collaborative frameworks using shared objects, in: *Proceedings of the 24th Australasian Conference on Computer Science,* Gold Coast, Queensland, Australia, 3–10.

Chan, S. C. F., Leung, C. W. K. & Ng, V. T. Y. (2003) GroupUML: a PDA-based graphical editor to support real-time collaboration in student group projects, in: *Proceedings of the International Conference on Computers in Education (ICCE 2003)*, 222–223.

Chang, C. Y., Sheu, J. P. & Chan, T. W. (2003) Concept and design of ad hoc and mobile classrooms, *Journal of Computer Assisted Learning,* 19(3), 336–346.

Chow, T. C. Y., Tsui, E. W. C., Chan, C. F. S. & Ng, V. T. Y. (2002) Wireless online tutoring system with collaborative diagramming, in: *Proceedings of the 2nd IEEE International Conference on Advanced Learning Technologies (ICALT 2002),* Kazan, Russia, 254–258.

Gardner, L., Sheridan, D. & White, D. (2002) A web-based learning and assessment system to support flexible education, *Journal of Computer Assisted Learning,* 18(2), 125–136.

Gokhale, A. A. (1995) Collaborative learning enhances critical thinking, *Journal of Technology Education,* 7(1).

Hammond, T. & Davis, R. (2002) Tahuti: a geometrical sketch recognition system for UML class diagrams, in: *Proceedings AAAI Spring 2002 Symposium on Sketch Understanding,* Palo Alto, CA, March, 25–27.

Handel, M. & Herbsleb, J. D. (2002) What is chat doing in the workplace?, in: *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work,* New Orleans, LA, 1–10.

Hansen, K. M. & Damm, C. H. (2002) Instant collaboration: using context aware instant messaging for session management in distributed collaboration tools, in: *Proceedings of the Second Nordic Conference on Human–Computer Interaction,* Aarhus, Denmark, 279–282.

Hron, A. & Friedrich, H. F. (2003) A review of web-based collaborative learning: factors beyond technology, *Journal of Computer Assisted Learning,* 19(1), 70–79.

Kreutz, R., Kiesow, S. & Spitzer, K. (2000) NetChat: communication and collaboration via WWW, *Educational Technology & Society,* 3(3), 87–93.

Landsberger, J. F. (2003) *Cooperative & collaborative learning.* Available online at: http://www.iss.stthomas.edu/studyguides/cooplearn.htm (accessed 25 September 2003).

Martin, J. & Odell, J. (1992) *Principles of object oriented analysis and design* (Englewood Cliffs, NJ, Prentice Hall).

Monson-Haefel, R. & Chappell, D. A. (2001) *Java message service* (Sebastopol, CA, O'Reilly).

Ratcliffe, M., Thomas, L., Ellis, W. & Thomasson, B. (2003) Capturing collaborative designs to assist the pedagogical process, in: *ITiCSE '03,* Thessaloniki, Greece, 30 June–2 July, 79–83.

Walker, D. W., Collings, P. & Richards-Smith, A. (2000) *Synchronous IT support for group work.* Available online at: http://simnotes.canberra.edu.au/synch.nsf?OpenDatabase (accessed 23 October 2003).

Wiersema, N. (2000) *How does collaborative learning actually work in a classroom and how do students react to it? A brief reflection.* Available online at: http://www.lgu.ac.uk/deliberations/ (accessed 25 September 2003).

Wolz, U., Palme, J., Anderson, P., *et al.* (1997) Computer-mediated communication in collaborative educational settings: report of the ITiCSE'97 working group on CMC in collaborative educational settings, *ACM SIGCUE Outlook,* 25(4), 51–68.