ONLINE SHARED EDITING FOR INTRODUCTORY PROGRAMMING COURSES

Hosung Song
Department of Information and Computing Sciences (Computer Science)
University of Wisconsin-Green Bay
2420 Nicolet Drive, Green Bay, WI 54311, U.S.A.

Keywords: Shared editing, Intro-programming courses, Pair programming, Remote instruction, Online learning, Remote collaboration.

Abstract: This paper argues how online shared editing can benefit teaching/learning intro-programming courses like CS1 and CS2. Various available tools supporting online shared editing are surveyed. A simple but novel Java-based online shared editing framework named *Olshed* is also presented and demonstrated. Olshed is a suite of Swing-based Java classes that supports easy development of online shared editing applications or extension of existing applications for online shared editing. As a proof-of-concept example, DrJava (an educational Java IDE) is extended with Olshed to provide real-time online shared editing facility.

1 INTRODUCTION

Real-time online collaboration has become very hot and important in the ubiquitously networked environment these days. In a real-time online collaboration environment, a group of users can edit a single document simultaneously and remotely using online real-time communication. This allows easier and more seamless collaboration between colleagues when they try to write a document collaboratively. Traditionally, they exchanged their shared documents as email attachments non-real-time, resulting in somewhat degraded collaboration performance. With the advent of real-time online collaboration technology, they can actually work on the same document at the same time even regardless of their geographic locations. In summary, online shared document-editing and collaboration service can be very beneficial and its future looks very promising.

In software development (especially coding), it is no longer a news that pair programming is one of the important aspects of "Extreme Programming" methodology (Beck, 2005), which is becoming fairly popular in many modern software development projects. In pair programming, two coworkers always code together in person simultaneously. One worker takes the role of "driver," entering all code through the keyboard. The other worker takes the role of "navigator" or "observer," reviewing each line of code as it is typed in. Many benefits of pair programming were identified and presented in (Cockburn and Williams, 2000). Pair programming technique has been suggested and tried for teaching/learning introductory programming courses too (Nagappan et al., 2003). By letting two students solve a coding problem using pair programming technique, students can even learn better from each other, on top of the original benefits of pair programming. However, many students these days have hard time to meet in person and program in pair due to their busy schedules and life styles. This makes the idea of "Remote Pair Programming" more appealing especially with ubiquitous availability of high-speed Internet access these days. Remote pair programming with online real-time shared editing can also be very useful in intro programming courses for other reasons like remotely assisting a student to identify simple syntax errors or tutoring of programming skills. No matter what situation two collaborators are in, a good tool support enabling remote pair programming is critical. One of the most important features of such a tool support is real-time online shared/collaborative editing of text document files.

In this paper, a number of available such tools are surveyed and compared in terms of their features (available or lacking), usability, and suitability for teaching intro-programming courses. Olshed (On-Line SHared EDiting), a simple Swing-based Java li-

Song H. (2010).
ONLINE SHARED EDITING FOR INTRODUCTORY PROGRAMMING COURSES.
In *Proceedings of the 2nd International Conference on Computer Supported Education*, pages 489-492
DOI: 10.5220/0002860904890492

brary for online shared editing developed by the author, is also introduced for the first time. Olshed is shown to be easily integrable with already existing Java applications, by demonstrating Olshed-extended DrJava (Allen et al., 2002), an educational Java IDE. The paper progresses as follows: Section 2 surveys select real-time online shared/collaborative editing tools. Section 3 explains the author's Olshed library and Olshed-extended DrJava is presented. Finally, section 4 concludes this paper by describing future plans for Olshed.

2 SURVEY OF EXISTING SHARED EDITING TOOLS

There have been quite many available existing real-time online shared editing tools. Most of them can be sorted into three categories: IDE-based, web-based, or stand-alone desktop applications.

2.1 IDE-based Shared Editors

IDEs (Integrated Development Environments) are indispensable these days for software development activities. Therefore, it is no surprise to see quite a few real-time online shared editing supports for widely used IDEs such as Eclipse. Especially, there are quite a few available Eclipse plugins supporting shared editing. Some notable ones are listed here with references:

- DocShare/Cola plugin in Eclipse Communication Framework (Lewis, 2007)
- Saros (Lau, 2009)
- XPairtise (The XPairtise Team, 2008)
- Sangam (The Sangam Project Team, 2008)

Most Eclipse plugins for shared editing utilizes some sort of central messaging server, especially a general purpose one such as XMPP/Jabber server. Once the plugin is installed, the first initiator can share her source code editor with another participant. Depending on the implementations, only two participants can share a document (ECF DocShare/Cola) or any number of participants can do so (Saros). Some Eclipse plugin like Saros allows the participants to be assigned a specific role (e.g. exclusive driver, shared driver, exclusive observer). Also plugins can differ from the number of files that can be shared (only one for ECF DocShare/Cola, an entire project with multiple source files for Saros). Overall, Eclipse plugins supporting shared-editing can be very useful if Eclipse is extensively used. However, the gravity of the solution (the heavy-weight nature of Eclipse itself and the need for a remote public messaging server or a local private server) can be an inhibitor for this approach. Shared editing over a publicly available remote messaging server can be pretty slow too.

2.2 Web-based Shared Editors

A web-based shared editing support can be viewed as a shared editor embedded in a web browser. Because a web browser cannot save and/or compile a source code file yet, users need to copy-and-paste collaboratively edited source code into an IDE and perform required compilation and debugging there. This could be a major inconvenience for intro-programming course purposes, but the ubiquity of web browsers and no need for installing anything are major attractions to this category. The following are somewhat well-known shared editing web applications/services:

- Etherpad[1]
- CollabEdit[2]
- MobWrite (Fraser, 2009)

All web-based shared editing applications heavily utilizes Javascript technology to make it possible to do real-time dynamic browser updates. All of them provide a very similar work flow. The first participant connects to the web site providing the shared-editing service, click a button for creating a public document (also called a pad) that can be shared-edited, and the returned URL is sent to other participants by email or instant messages. The later participants just need to browse the delivered URL to join the shared-editing session. Etherpad provides probably most advanced features such as displaying each individual participant's entry in a different color (called Authorship Colors) and the Time Slider feature allowing any participant to navigate through all the revision history of the shared document.

2.3 Stand-alone Desktop Shared Editors

Stand-alone desktop shared editor applications may not have a firm ground these days due to the recent proliferation of IDE and browser-based applications. However, when a remote server for sharing is not desired or a light-weight application is desired instead of a heavy IDE, a simple stand-alone shared editor application could be a good choice. Here are some select stand-alone desktop shared editing applications.

[1] http://etherpad.com/
[2] http://collabedit.com/

- Notepad++ with NetNote plugin (The Notepad++ Project Team, 2009)
- ACE: A Collaborative Editor (ACE Project Team, 2006)
- Gobby (0x539 Dev Group, 2009)
- SubEthaEdit[3]

There are actually quite a lot more than the ones selected above. For more complete list, readers are refer to http://en.wikipedia.org/wiki/Collaborative_real-time_editor. Usage-wise, stand-alone desktop shared editing applications are not too different from the other two categories, except two points. Firstly, some applications are not cross-platform (Notepad++ is Windows-only and SubEthaEdit is Mac-only) due to the adopted development frameworks. Secondly, networking can be just one-to-one (Notepad++) or many-to-many with auto-client discovery protocols such as Bonjour (SubEthaEdit and ACE). They are mostly pretty light-weight, but their suitability for beginning students in programming is not the best yet.

3 OLSHED: A SWING-BASED JAVA LIBRARY FOR ONLINE SHARED EDITING

As surveyed above, there are lots of available existing real-time online shared-editing applications (either desktop or web-based). However, CS1/2 students or instructors may have hard time to pick one suitable for their needs. Eclipse plugins may not be feasible if Eclipse is not used for a course due to its heavy weight. Even though Eclipse is used, installing and using additional plugins can be difficult for beginning CS1/2 students. Light-weight desktop or web-based shared editor applications lack the integration with compiler and debugger. Therefore, it can be argued that enabling a light-weight IDE (e.g. DrJava or BlueJ) with a simple shared-editing feature would be highly desirable. Since most light-weight IDEs are also written in Java with Swing GUI framework, it is natural to develop a Swing-based Java Library for shared editing.

Olshed (OnLine SHared EDiting)[4], developed by the author and presented here for the first time, is a suite of Java classes to easily extend any Swing-based Java applications with real-time online shared editing feature. Communication is done by a simple multi-threaded Java chat server with custom protocol mes-

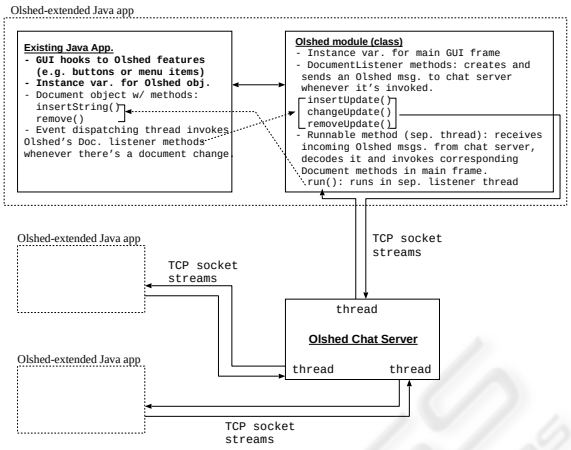

Figure 1: Olshed Architecture.

sage formats through standard Java I/O streams. Olshed library utilizes Java's Swing framework's DocumentListener[5] and Document[6] interfaces.The Olshed main class implements DocumentListener interface implementing three update methods (change, insert, and remove) and an instance of the Olshed main class is added as a DocumentListener to the associated Document object of the target Swing Java application that is extended for online shared editing. This way, every document update on the target application's document model will be notified to the Olshed document listener code. Upon being notified, the Olshed document listener code creates and sends a message to the Olshed chat server through the output stream of the socket connection established earlier. The Olshed chat server relays the message to all connected Olshed clients. Incoming messages are handled by a separate chat-listener thread through the input stream of the established socket connection. The Olshed chat-listener thread (in the client) then invokes the corresponding update method (insertString(), remove(), or createPosition()) on the registered Document object from the target application. Figure 1 shows the overall architecture of Olshed.

Necessary modification of the target application's code is minimal thanks to the full utilization of Swing framework. The target application's code needs to be extended with necessary GUI hook(s) to Olshed features (like buttons or menu items) and an instance of the Olshed main class, which just needs to be instantiated in the constructor of the target application's main GUI class. The Olshed main class's constructor also initiates a socket connection to the Olshed chat server

[3] http://www.codingmonkeys.de/subethaedit/

[4] Available at http://icsa.uwgb.edu/ songh/olshed/ as open source software.

[5] http://java.sun.com/docs/books/tutorial/uiswing/events/documentlistener.html

[6] http://java.sun.com/j2se/1.5.0/docs/api/javax/swing/text/Document.html

(the connection parameters should be passed to the constructor) and starts a chat-listener thread. The instantiated Olshed object should later be told about the actual Document object on which shared editing will be conducted. At that time, the Olshed main class's instance is also added to the Document object as a DocumentListener object.

As a proof-of-concept realization of Olshed, DrJava (Allen et al., 2002), a well-known educational Java IDE, was extended with Olshed for real-time online shared editing. It was easily done by following the simple extension strategy described above, without serious modification to DrJava's original code base. Any number of DrJava+Olshed application instances can perform shared editing of a single Java source code file document as long as they are all connected to the same Olshed chat server. This is a strong evidence that Olshed can be easily applicable to any existing Swing-based Java applications.

The development of Olshed is still in its infant stage. Currently, the document consistency maintenance algorithm is yet to be implemented. The lack of the feature can possibly result in out-of-sync documents across Olshed clients at this point. As long as the current infantile Olshed is used in a purely driver-and-observer model, this shouldn't be a big issue for the time being. Currently Olshed can share only a single document among two or more participants. The current Olshed chat server implementation allows only one group of shared editing participants. Lifting these restrictions should not be difficult.

4 CONCLUSIONS

In this paper, it was informally argued that real-time online shared editing can significantly benefit not only general remote collaboration like pair programming but also teaching and learning of introductory programming courses. Available existing online shared applications were surveyed. Hoping a light-weight IDE-based shared editing environment for use by students and instructors of intro-programming courses, the author developed and presented Olshed, a simple Swing-based Java library allowing an easy extension of existing Swing Java applications with real-time online shared editing. As a proof-of-concept project, DrJava, a well-known Java IDE written in Java, was extended for shared editing with Olshed. The full source code of Olshed and Olshed-extended DrJava is available as an open source project.

There are many needed future work topics for Olshed. The most imminent feature that's needed is to maintain document consistency under concurrent write access to a single document. An OT (Operational Transformation)-based algorithm (Sun and Ellis, 1998) will be soon implemented and integrated. Other desirable features would be related to the Olshed chat server, such as supporting multiple concurrent editing groups and supporting multiple documents editing per each concurrent editing group.

REFERENCES

0x539 Dev Group (2009). Gobby. http://gobby.0x539.de/.

ACE Project Team (2006). Ace - a collaborative editor. http://sourceforge.net/projects/ace/.

Allen, E., Cartwright, R., and Stoler, B. (2002). Drjava: A lightweight pedagogic environment for java. In *Proceedings of ACM SIGCSE Symposium 2002*.

Beck, K. (2005). *Extreme Programming Explained - Embrace Change*. Addison-Wesley, 2nd edition.

Cockburn, A. and Williams, L. (2000). The costs and benefits of pair programming. In *Proceedings of the 1st International Conference on Extreme Programming and Flexible Processes in Software Engineering (XP2000)*.

Fraser, N. (2009). google-mobwrite. http://code.google.com/p/google-mobwrite/.

Lau, S. (2009). Saros - Distributed Pair Programming for Eclipse. https://www.inf.fu-berlin.de/w/SE/DPP.

Lewis, S. (2007). RT Shared Editing - Eclipsepedia. http://wiki.eclipse.org/RT_Shared_Editing.

Nagappan, N., Williams, L., Ferzli, M., Wiebe, W., Yang, K., Miller, C., and Balik, S. (2003). Improving the cs1 experience with pair programming. In *Proceedings of ACM SIGCSE Symposium 2003*.

Sun, C. and Ellis, C. (1998). Operational transformation in real-time group editors: Issues, algorithms, and achievements. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work*.

The Notepad++ Project Team (2009). Notepad++. http://notepad-plus.sourceforget.net/.

The Sangam Project Team (2008). Sangam - Eclipse Plugin. http://sangam.sourceforge.net/.

The XPairtise Team (2008). XPairtise - Pair Programming for Eclipse. http://xpairtise.sourceforge.net/.