CodeAnnotator: Digital Ink Annotation within Eclipse

Xiaofan Chen
Auckland University
Auckland, New Zealand
xche044@ec.auckland.ac.nz

Beryl Plimmer
Auckland University
Auckland, New Zealand
beryl@cs.auckland.ac.nz

ABSTRACT
Programming environments do not support ink annotation. Yet, annotation is the most effective way to actively read and review a document. This paper describes a tool, CodeAnnotator, which integrates annotation support inside an Integrated Development Environment (IDE). This tool is designed and developed to support direct annotation of program code with digital ink in the IDE. Programmers will benefit from a more intuitive interaction space to record notes and comments just as they would on paper documents.

ACM Classification Keywords
H5.2 [Information interfaces and presentation]: User Interfaces, Graphical user interfaces.

General Terms
Design, Performance, Experimentation

Author Keywords
Digital ink annotation, annotation, sketch, eclipse

1. INTRODUCTION
Annotating documents when reading assists readers by enhancing reading-comprehension activities [5, 6]. Moreover, reading documents that have been annotated by previous readers improves subsequent readers' understanding and recall of the emphasized items [15, 18]. People are accustomed to making annotations with pens on paper documents. More recently, researchers have been exploring adding digital ink functionality to provide annotation over digital documents [8, 9, 11, 12]. However, Integrated Development Environments (IDE) do not support digital ink annotation. People cannot make annotations directly inside the IDEs without the addition of annotation support.

This project aims to provide people with an electronic environment to annotate code documents within an IDE. We designed and developed a tool, CodeAnnotator, to support users directly annotating over code within Eclipse. As this tool integrates annotation support inside the IDE, it offers additional advantages inherent from typical code review processes, as explained by Priest [12].

With CodeAnnotator after loading a code file in an IDE, users can write annotations using a digital pen while also having all the functions offered by the IDE, such as debugging, compiling, and executing. Therefore, users have both IDE and annotation support.

This paper begins with a summary of related work on digital ink annotation. Then it presents the design requirements and implementation of CodeAnnotator, followed by conclusions and future work.

2. RELATED WORK
Most early annotation tools, such as Wang Freestyle [3] and XLibris [4, 14], only supported annotating over static documents. These tools provide users with simple free-form ink annotation and an interface and features similar to that of paper. They are self-reliant, independent from environments and may be used by people when editing and reading digital documents.

With the experiences researchers gained from development of self-reliant annotation tools, new tools that integrate annotation support inside other applications have emerged. For instance, people can annotate over documents directly in Microsoft Word [7] and MADCOW [2], WPM [17] and Web annotation tool [13] assist users in making annotations in web browsers.

However, code documents have a unique feature that other types of documents do not have: code is non-linear; it is arranged in logical classes and procedures that are not intended to be read sequentially like a book [12]. Some digital ink annotation tools have been designed and developed to meet the unique features of code. Penmarked [11] is a tool for annotating and marking student assignments on Tablet PCs. Teachers can read the assignments and add digital ink annotations directly on the document. The limitation of this system is it only deals with static documents and opens files as a text file, so the code (Java, C# etc.) can not be compiled or run under this system. Gild [10] is a set of plug-ins for Eclipse to mark assignments written in Java, however, it doesn't support digital ink annotation. RCA [12] is a code annotation tool added to Visual Studio .Net 2005 (VS). Users record digital ink annotation on a code file opened in VS, and also use all the functions provided by VS. However, it doesn't provide navigation of annotations to support users finding and moving between annotations.

A program document is normally very long and split among many files. Navigation support of annotations can provide users with an outline of existing annotations and the information of how many and where annotations exist. In addition, it assists users to easily locate a specific annotation. Among the current available annotation tools, there are no tools that provide navigation support.

OzCHI 2007 Proceedings, ISBN 978-1-59593-872-5

3. DESIGN

Digital ink annotation requires support for three major functions: the annotations must be free-form and modifiable, the annotations can be reflowed when the underlying text is changed, and navigation support is provided to help users to overview, select and find annotations.

3.1 Requirements

The IDE must be extensible to allow developers to extend its functionality and add in other functions. In order to allow the IDE to perform normally while supporting digital ink annotation, CodeAnnotator must integrate into the IDE seamlessly.

When a user first selects a code file to annotate, a new annotation file will be created and stored as a part of the development project. Once the user finishes the annotation, the annotation file needs to be saved to allow later review. The annotation file will be loaded when the user wants to review or annotate the selected code file.

3.2 Free-form and modifiable annotation

Free-form annotation consists of words, symbols and text selection marks, and allows annotations to be made anywhere without limitation on shape or content [14]. This is achieved by overlaying the code window with a transparent canvas that holds ink annotations and associating annotations to the underlying code by making the code window and the transparent canvas scroll together. This approach allows users to annotate anywhere inside the code window; there is no restriction of a pagination boundary in the code window as there is in document formats such as PDF that set artificial restraints on the users' positioning of the annotations

When people annotate over paper documents with pens, it is hard to modify existing annotations. Modifying real ink annotations usually involves concealing or crossing out the annotations then rewriting them, this looks messy. A digital ink annotation tool provides users with a means to efficiently and cleanly erase, select, move and recolor a selected annotation.

3.3 Reflowing annotations

One of the most important features of digital ink annotation tools is to be able to deal with dynamic digital documents. As annotation is most often a function of some type of review it must be expected that the text inside the documents will be changed. This requires the existing annotations reflow to remain consistent with the underlying code when the code file is modified. In order to reflow annotations, ink strokes have to be grouped. Annotations belonging to the same underlying context must be grouped together by taking into consideration location and temporal properties of the ink strokes that make up the annotation [1, 16]. Then when underlying code moves up/down, any attached annotation group has to move too. And when underlying code is deleted, its attached annotation must either be deleted or archived.

A code file is arranged line by line. Each code line cannot be rearranged like a sentence and typically lines are not wrapped when resizing the code window. Therefore, it is unnecessary to consider the issue of reflowing annotations horizontally. We only need to focus on moving annotations up or down inside the code window.

3.4 Navigation support

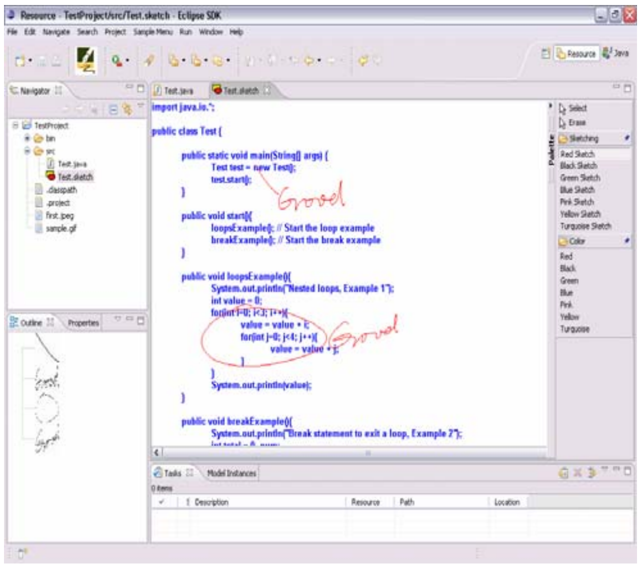
A project contains many code files. A code file is structured by its classes and procedures. IDEs often offer a navigation system to assist programmers to know which code files exist in the project and how classes are related in a code file, and to locate a specific class.

For digital ink annotation tools, it is useful to provide a similar navigation system to users. A code file often extends over many lines and cannot be fully displayed in one screen. We have to scroll the screen up/down to review existing annotations, which is very inconvenient. Through the navigation system, users can easily locate a specific annotation inside the code window.

4. IMPLEMENTATION

CodeAnnotator is developed inside Eclipse as a set of plug-ins. Because Eclipse supports Java language it is a popular open source IDE that is used extensively in teaching departments and industry.

CodeAnnotator can be used on Tablet PCs or desktop computers with a tablet USB input device. It allows users to directly annotate onto the output screen. This tool is implemented in Java using both the Java API and Eclipse framework.



```
import java.io.*;

public class Test {

        public static void main(String[] args) {
                Test test = new Test();
                test.start();
        }

        public void start(){
                loopsExample(); // Start the loop example
                breakExample(); // Start the break example
        }

        public void loopsExample(){
                System.out.println("Nested loops, Example 1");
                int value = 0;
                for(int i=0; i<3; i++){
                        value = value + i;
                        for(int j=0; j<4; j++){
                                value = value + j;
                        }
                }
                System.out.println(value);
        }

        public void breakExample(){
                System.out.println("Break statement to exit a loop, Example 2");
```

Figure 1. CodeAnnotator in Eclipse

Figure 1 shows an annotation window in Eclipse. When a user loads a code file and select annotation model from the Eclipse toolbar, a new *annotation window* is created based on the current code window and an *outline window* is created. The flyout-Palette at the right side of the annotation window contains the annotation tools.

The major features of CodeAnnotator are: attaching annotations to a specific code line, grouping annotations and editing them, reflowing annotations, and navigating annotations.

4.1 Attaching Annotations

Ink strokes are used as linkers to link a group of annotations to a specific code line [12]. Linkers are a line or a circle. Each is treated differently so first we have to recognize the linkers. A feature of a straight line is that its start and end points touch the left and right border of the bounding box (See Figure 2). A circle can be discriminated from a line by measuring whether the distance between the first and last point (ab) less than 25% of the hypotenuses of the bounding box (a'b') (See Figure 3).

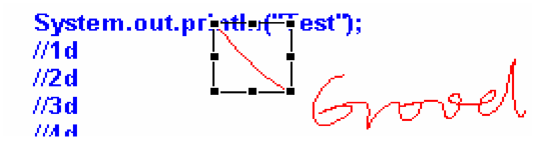


Figure 2. Line linker

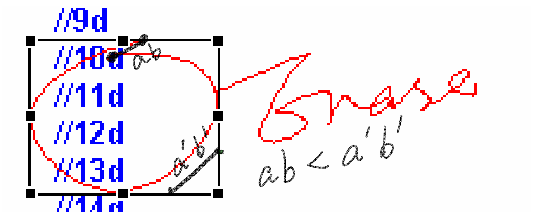


Figure 3. Circle linker

If the linker is a line, then its corresponding annotation group is linked to the code line closest to the start point of the linker. Otherwise, for a circle the corresponding annotation group is attached to the line closest to the middle point of the bounding box.

4.2 Group and edit annotations

We group annotations based on the spatial and temporal properties of the annotations [1, 16]. After creating a linker, if the subsequent annotation is created in less than two seconds, then it belongs to this linker. If the following annotation is created in less than two seconds, then it also belongs to this linker. Otherwise, we use the spatial property of the annotation to decide which group it belongs to.

Spatially adjacent annotations are considered to be in the same group. We calculate the position of the new annotation; say at the coordinate (x1, y1), its height is h1, its width is w1. According to this calculated position, we find the closest annotation group and then this annotation belongs to this group. For example, the first group is located at the coordinate x2, y2, and size h2, w2 and the second group is located at the coordinate x3, y3, and size h3, w3. (See Figure 4)

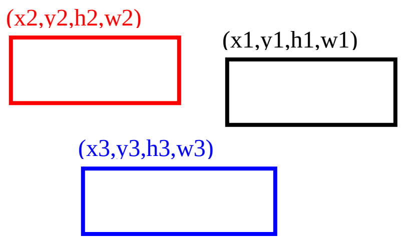


Figure 4. Group annotations

- If the range from y1 to y1+h1 partially overlaps with the range from y2 to y2+h2, then the new annotation belongs to the first group.
- If the range from y1 to y1+h1 partially overlaps with the range from y3 to y3+h3, then the new annotation belongs to the second group.
- If the range from y1 to y1+h1 overlaps with both the range from y2 to y2+h2 and the range from y3 to y3+h3, then we have to decide which is closest:
    - If x1 is bigger than x2 and x3, then if x1 is closer to x2+w2 than to x3+w3, then the new annotation belongs to the first group, otherwise it belongs to the second group.
    - If x1 is less than x2 and x3, then if x1 is closer to x2 than to x3, the new annotation belongs to the first group, otherwise it belongs to the second.
    - If x1 is bigger than x2 and is less than x3, then x1 belongs to the first group because we are more likely to append comments to the existing annotation.
    - If x1 is bigger than x3 and is less than x2, then x1 belongs to the second group
- Otherwise, we consider the new annotation to be a new group.

Editing annotations is a very important function in annotation tools. We support moving a specific annotation group around by selecting and then dragging it, the linker position is recalculated after a move. Also an annotation can be deleted or erased by selecting it with the eraser tool. Annotations can be written in different colors by selecting a color before writing. Moreover, we can recolor an existing annotation by selecting it and then selecting a different color from the color tool.

4.3 Reflow annotations

Reflowing existing annotations is necessary to maintain consistency with the underlying code. We only need to consider the vertical reflow as code is line-based. There are three situations we need to handle, as explained by Priest [12]. First, when several code lines are added ahead of a code line with annotations, this code line and its associated annotation group moves down together. Second, when several code lines are deleted ahead of a code line with annotations, this code line and its corresponding annotation group moves up together. Third, when a code line with annotations is deleted, its associated annotation group is also deleted.

4.4 Navigation support

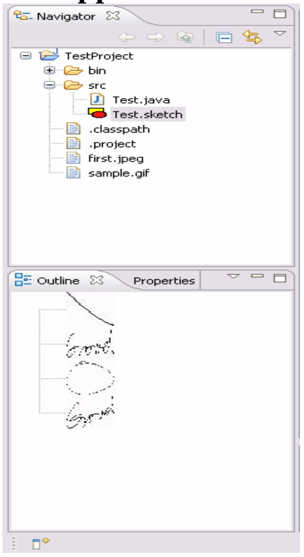


Figure 5. Navigation support

The navigation system includes a Navigator window and an Outline window. The Navigator window tells users which code files were annotated. The Outline window guides users to locate the position of a specific annotation inside the annotation window, and provides a graphical outline of existing annotations. Users can directly open an annotation file in the Navigator window. After this file is opened, an Outline window is automatically opened and shows a graphical outline of existing annotations in this file. When users make an annotation in the annotation window, a copy of this annotation and its corresponding information such as location will be put inside the Outline window (See Figure 5). When users select a specific annotation inside the Outline window, the annotation window will scroll to the page containing this annotation.

5. CONCLUSIONS AND FUTURE WORK

This paper describes a code digital ink annotation tool, CodeAnnotator. The objective of this tool is to provide users with an electronic environment to annotate in Eclipse. This tool lets users enjoy both digital ink annotation and Eclipse support. In other words, after loading a code file in Eclipse, users can write annotations with a digital pen and can also access all the functions provided by Eclipse, such as debugging, compiling, and executing.

Including handwriting recognition is a challenge because the writing and iconic annotation must first be separated. We are yet to extend CodeAnnotator to recognize handwriting. When this is completed the next steps are to conduct usability testing and evaluation studies to assess the efficacy annotation within an IDE.

Another interesting extension would be functionality to consolidate digital ink annotations made by different users into one file to support collaborative code review. The separate digital ink annotations can be merged automatically with some mechanism to differentiate each person's annotations, particularly when they are in the same place. Furthermore, we could extend CodeAnnotator to support assignment marking. This would allow markers to directly mark and comment programming assignments in Eclipse.

6. REFERENCES

[1] Bargeron, D. and Moscovish, T. Reflowing digital ink annotations. *CIII 2003, April 5-10*, ACM Press (2003),385-392

[2] Bottoni, P., Levialdi, S., Labella, A., Panizzi, E., Trinchese, R. and Gigli, L. MADCOW: a visual interface for annotating web pages. *Proc. of the working conference on Advanced visual interfaces AVI '06*, ACM Press (2006), 314-317

[3] Francik, E. Rapid, Integrated design of a multimedia communication system. *Human Computer Interface Design (1995)*, 36-69

[4] Golovchinsky, G. and Denoue, L. Moving Markup: Repositioning freeform annotations. *Proc. UIST 2002*, ACM Press (2002), 21-30

[5] Marshall, C. C. Annotation: from paper books to the digital library. *Proc DL 1997*, ACM Press (1997), 131-140

[6] Marshall, C. C. Toward an ecology of hypertext annotation. *Proc. HyperText 1998*, ACM Press (1998), 40-48

[7] Microsoft Word.

http://www.microsoft.com/office/word/using.htm

[8] Mock, K. Teaching with Tablet PCs. *Proc. Journal of Computing Sciences in Colledges 20, 2 (2004)* 17-27

[9] Moran , T. P., C, P. and Van Melle, W. Pen-based interaction techniques for organizing material on an electronic whiteboard. *In Symposium on User Interface Software and Technology (1997*), 45-54

[10] Myers, D., Hargreaves, E., Ryall, J., Thompson, S., Burgess, M., German, D. and Storey, M. Developing marking support within Eclipse. *Proc. Of OOPSLA 2004, ACM Press (2004)*, 62-66

[11] Plimmer, B. and Mason, P. A Pen-based paperless environment for annotating and marking student assignments. *Proc. 7th Australasian User Interface Conference*, CRPIT press (2006), 37-44

[12] Priest, R. and Plimmer, B. RCA: Experiences with an IDE annotation tool. *Proc. 6th ACM SIGCHI New Zealand chapters international conference 2006*, ACM Press (2006), 53-60

[13] Ramachandran, S. and Kashi, R. An architecture for ink annotations on web documents. *Proc. 17th International Conference on Document Analysis and Recognition*, IEEE Computer Society (2003), 256-260

[14] Schilit, B. N., Golovchinsky, G. and Price, M.N. Beyond Paper: Supporting active reading with free form digital ink annotations. *Proc. CHI 98*, lOS Angeles, CA, ACM Press (1998), 249-256

[15] Schumacher, G. M. and Nash, G. J. Conceptualizing and Measuring knowledge change due to writing. *Research in the teaching of English, vol 25*, pp. 67-96, 1991

[16] Shilman, M. and Viola, P. Spatial recognition and grouping of text and graphics. *EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling 2004*. Eurographics digital library

[17] Takahiro, K., Tashiro, N., Ozono, T., Ito, T., and Shintani, T. Web Page Marker: a web browsing support system based on marking and anchoring. *WWW 2005, May 10-14*. Chiba. Japan. ACM, 1012-1013

[18] Wolf, J. L. Effects of annotations on student readers and writers. *Digital Libraries, San Antonio*, TX, ACM Press (2000) 9-26