SKETCH UML:
A TABLET PC-BASED E-LEARNING TOOL FOR UML SYNTAX USING A MINIMALISTIC INTERFACE

Bastian Tenbergen, Colleen Grieshaber, Lisa Lazzaro, & Rick Buck
Human-Computer Interaction MA Program

The Interactive Learning Technology Laboratory of the State University of New York at Oswego has developed a Tablet PC-based, ink-input enabled learning utility that will aid students in learning the UML syntax as well as teachers in critiquing UML diagrams designed by students. This is intended to enhance the learning and teaching experience regarding UML using non-traditional interfaces. As a merely ink-based application, SketchUML is a novel approach in pen-based user interfaces. As such, the usability of it has not been assessed. This study addresses the question of usability of non-hybrid, ink-only, non-traditional interfaces by conducting usability tests. The collected data shows that ink-only interfaces facilitate learning in non-expert users and do not hinder the usability and learnability of these interfaces and allow for an enhanced learning experience.

I. Introduction

SketchUML was conceived as an interactive learning utility that allows for learning the syntax of the UML modeling language using Tablet PCs. UML – or Unified Markup Language – is a general-purpose standardized specification modeling language, used in Software Engineering to model object diagrams and abstract specifications of software systems. UML has a wide range of symbols and syntactical rules according to which valid diagrams are designed (Fowler, 2003). Learning the entire UML syntax can be tedious and time consuming, due to its error-driven nature.

Computer-aided UML learning has a number of advantages over ordinary approaches using pen and paper. The fact that it allows for immediate feedback to the user if a certain UML symbol is incorrect in the current context is an invaluable tool that cannot be accomplished by pen and paper approaches. Furthermore, computer-aided UML design enables the user to modify an existing diagram quickly without the need to redesign a specific section or even the entire diagram. Another advantage that is provided by incorporating software into UML diagram design is that from an existing class diagram, code can be exported, resulting in class and method stubs in a programming language of choice that obey to the class hierarchy depicted in the diagram and only have to be filled with content.

However, using software to design UML schemes has a negative impact on the learning curve of UML. When a student is asked to learn the UML syntax by using a certain software, this generally results in the fact that the software has to be learned beforehand. It is very uncommon for a user to know how to use a software that performs a specific task without having in-depth knowledge of the task itself. Many

UML softwares that are currently available do not tackle this problem. They are not designed to facilitate the learning experience of a user to learn the syntax of UML. Instead, these programs are designed to incorporate more and more features, which results in amazingly complex software that is hard to use for novices.

This problem is tackled by SketchUML. Previous publications by Qiu (2007) have shown that this software is a very usable and versatile tool for learning and designing UML diagrams. The product has been massively augmented since Qiu (2007) and incorporates many new features. SketchUML is a completely user centered software. It is designed to facilitate the learning of computer-aided UML design, so that it is virtually entirely naturalistic. The interface is designed with simplicity being the driving force, offering only a paper-like white canvas and a menu bar. Users can simply open the application and start drawing UML diagrams, as if they would use pen and paper. Designed for Tablet PCs, SketchUML accepts input using the Tablet PC's pen or stylus and converts the drawn ink into UML components, if the ink input corresponds to a valid symbol in UML. This allows for immediate feedback to the user when an ink gesture is not valid UML syntax component. The user is thereby forced to design UML diagrams accurately which aids in learning the UML syntax. Since SketchUML is designed to be as versatile as possible, it allows to fully edit the entire diagram at any point in the design process, just as users would expect to use pen and paper. Also, SketchUML is designed to be robust against different drawing styles of ink gestures that represent a UML symbol – for instance, it does not matter if a user draws a square by drawing one side at a time, lifting the pen tip every time a stroke has been performed, or if the user draws a square by using only one stroke from start to end, with the start and end point being identical.

Previous studies have shown that using Tablet PC technology is an effective tool for teaching and collaborative learning (Bull et. al., 2004; Berque et. al., 2004; Simon et al., 2004). SketchUML is intended to allow for an augmented teaching experience for faculty teaching UML to students. This is achieved by the ability to import a student's diagram that has previously been saved to hard disk, and critiquing it using the Teacher Mode. Critiquing the students' diagrams allows the teachers to give feedback on the student's work online, without the need of paper. Also, this reduces the amount of time needed to correct students' solutions to a given assignment. More significantly, it aids the teacher in a way that there is no need for an actual teacher's model solution. It is conceivable that the system can create a model solution or a solution template by analyzing a number of student solutions that have been critiqued by the teacher.

Overall speaking, SketchUML offers a good learning platform for students and teachers with regard to UML diagram design and syntax learning. Its wide variety of user centered features and the incorporated design metaphors create a well developed conceptual model that allows users to learn the syntax of UML rather than learning the software that is supposed to aid them to do so first. This paper discusses the research that has been done to augment the usability of SketchUML. A number of usability tests have been conducted in order to assess the pitfalls of ink-only, non-traditional interfaces and to understand how these types of interfaces can aid in e-Learning. The results of these experiments directly influenced the understanding of how learners interact with learning software and how the proposed software SketchUML needs to be augmented in order to provide for a decreased learning curve and a more naturalistic approach in Advanced Learning Technologies.

II. Methods

The main purpose of the tests was to assess the limitations of ink-enabled user interfaces. SketchUML is not only an e-learning tool that facilitates learning of the UML syntax, it also a novel approach for only ink-input accepting computer interfaces. The testing session was intended to identify the limitations of ink-only input and evaluate the usability of non-hybrid interfaces (rather than hybrid ink & point-and-click-based interfaces) and the effectiveness of SketchUML as an e-Learning tool for UML diagrams.

II.a. Participants

A number of undergraduate students (7 seniors, 3 juniors and 1 sophomore) that were enrolled in a software engineering class have been tested in this study. The students were all novices to UML or had intermediate knowledge, but no student was an UML expert. The participants' experience with computers and user interfaces ranged from 5 to 15 years (M = 12,8, SD = 2.5). No student received any reward for participating in the study and their participation was voluntary. All but two students were new to Tablet PCs and the SketchUML software.

II.b. Apparatus

The environment was a classroom setting with other students, desks and typical supplies. The lighting was also that of a typical classroom. The equipment being used was 6 Hewlett Packard Compaq TC4200 Tablet PC Laptops, with 1.7GHz Intel Centrino Processors, 512MB Ram, running Windows XP Tablet PC Edition. The test software was the latest stable release of SketchUML. For testing, the laptops have been switched to Tablet Mode and participants were asked to use the Pen as the input device.

II.c. Procedure

After the participants signed informed consent forms and filled out a pre-test demographic questionnaire, they were given a short introduction to the interaction with Tablet PCs and the software SketchUML. The introduction consisted of a brief overview of the features of SketchUML and the features of the Tablet PC hardware that was used and was presented using a standard classroom projector. All participants were introduced at the same time so every participant received the same introduction. An effort was made to randomly divide the participants into two groups, so that 6 participants were in group one and 5 participants were in group two. Each participant was given a task sheet with a task description; the task the students had to complete was different for both groups. The members of group one were given a class hierarchy description and were asked to draw the class hierarchy using the Tablet PCs, on which SketchUML was running. Group two was given the same class hierarchy description and a pre-loaded class diagram that was drawn incorrectly and does not meet the hierarchy description. The erroneous parts of the diagram were labeled accordingly (e.g. The class name “wrong class” for a class that must be deleted), with exception of incorrectly placed connectors, as connector labeling support is currently not fully supported in SketchUML. Participants in group two were asked to critique the pre-loaded diagram using the Teacher Mode that is built into SketchUML. Fig. 1 shows the experimenters ideal solution to the task of the first group. Fig. 2 shows the pre-loaded diagram the participants in the second group were asked to critique and Fig. 3 shows the ideal solution to the task of group two.

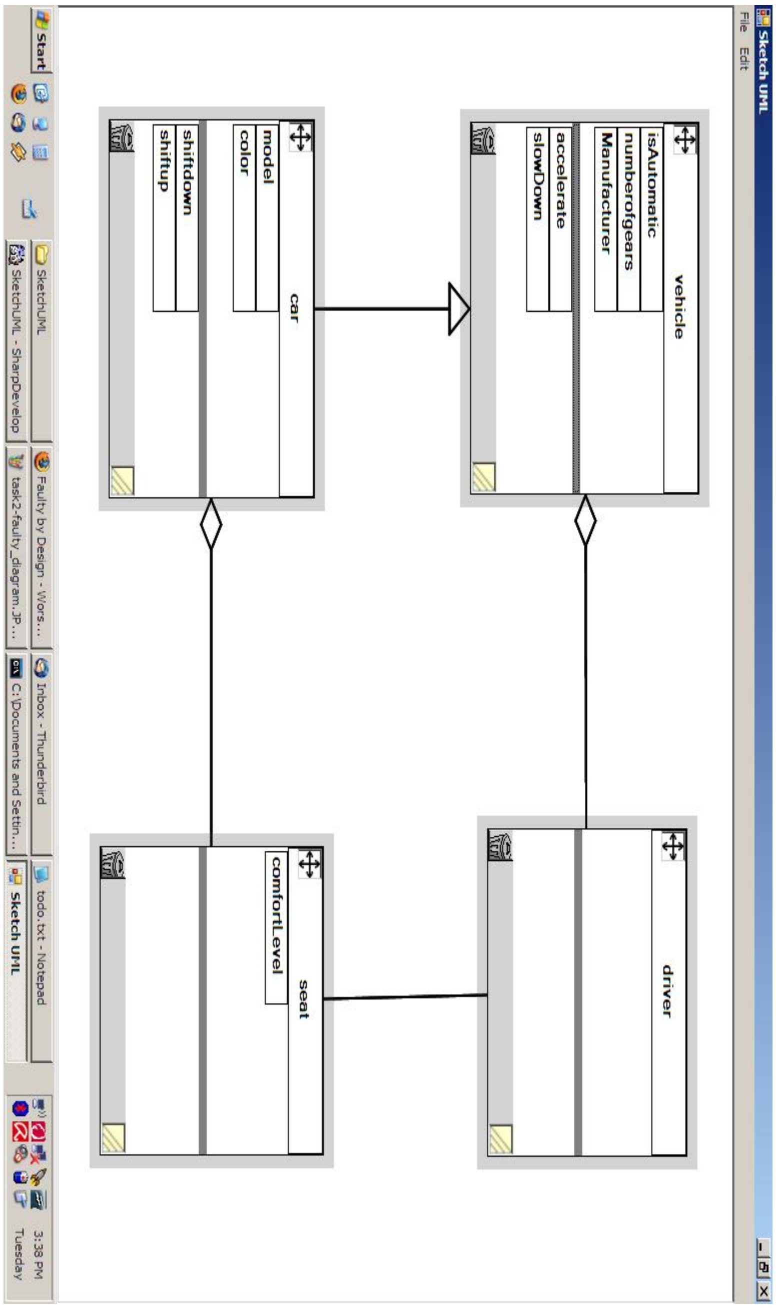

Fig. 1. Ideal Solution to the Task of Group One

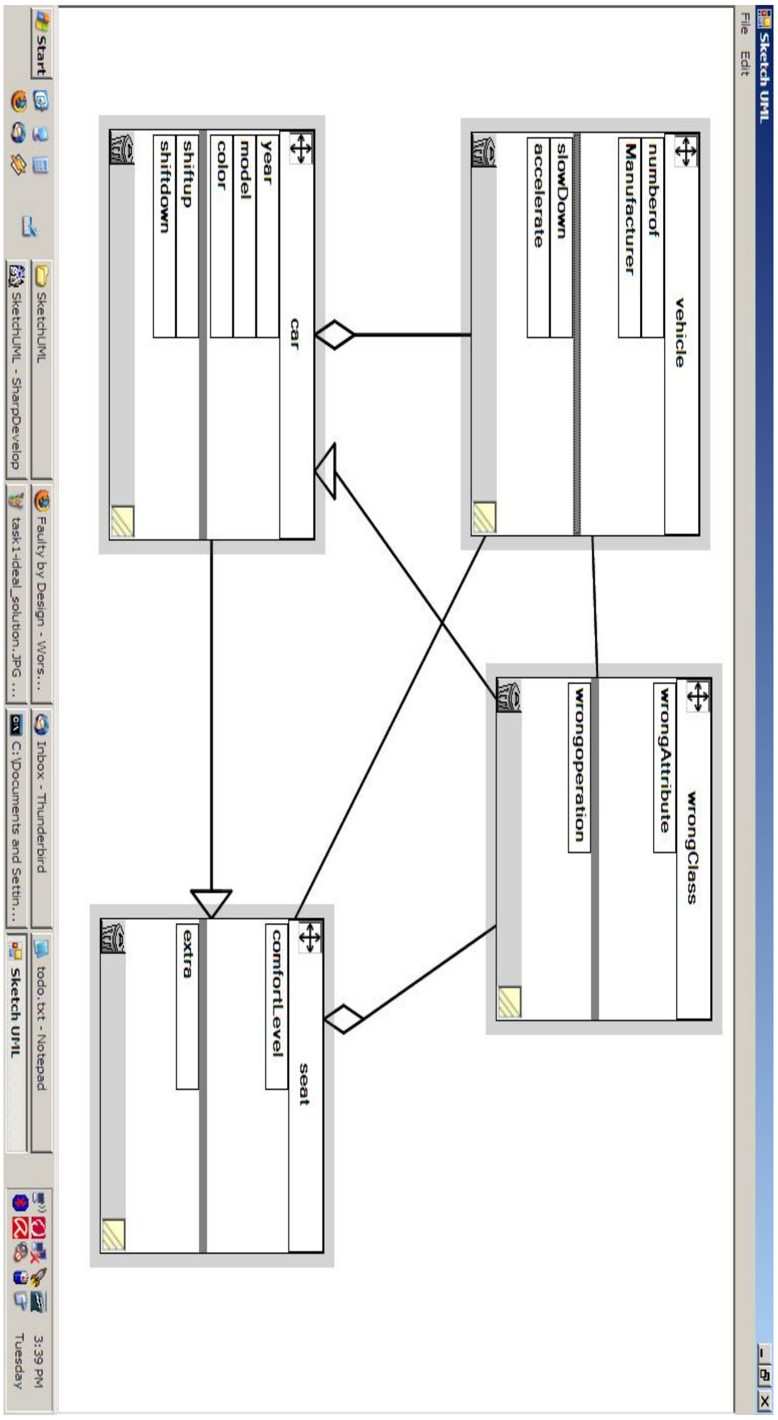

Fig. 2. Pre-Loaded Diagram to the Task of Group Two

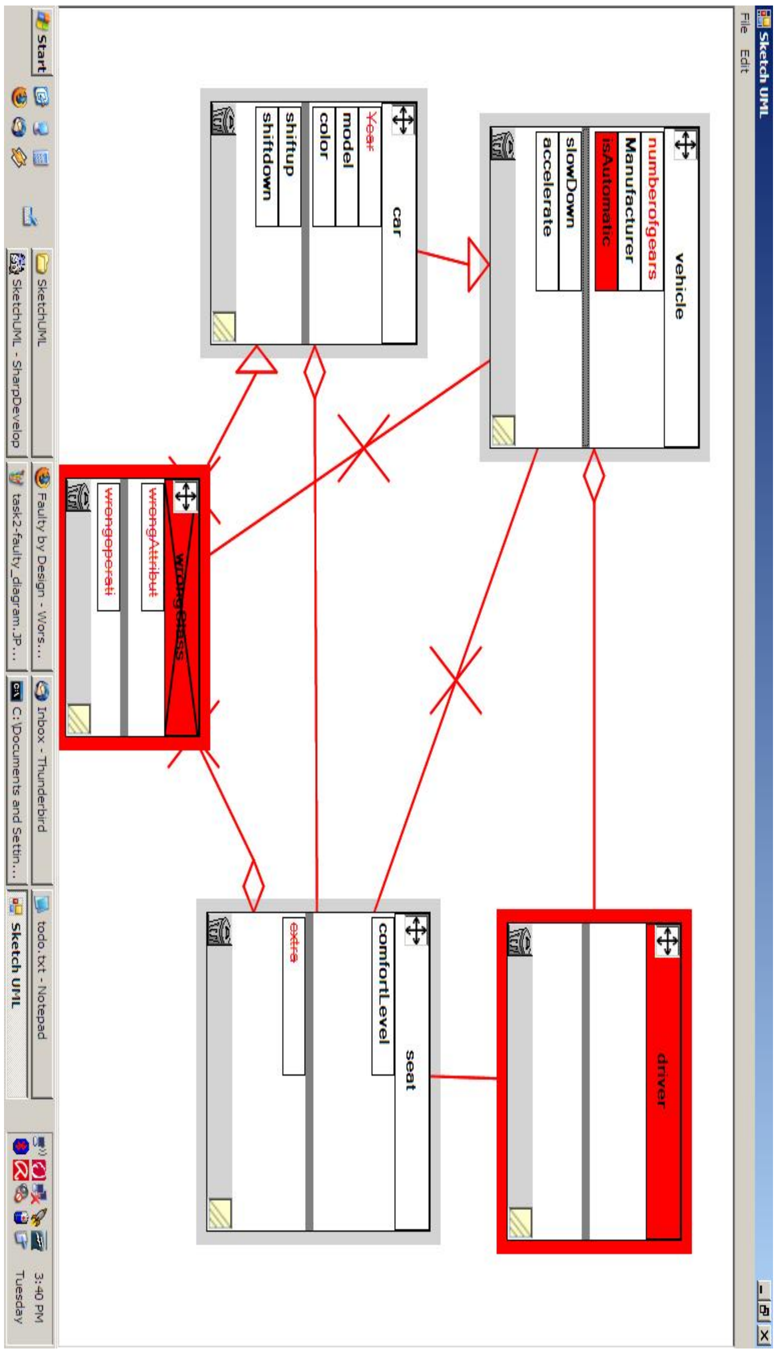

Fig. 3. Ideal Solution to the Task of Group Two. Note, that some elements are shown in a red background in the original program

The tasks were designed so that every feature of SketchUML was used at least once by at least seven participants throughout the experiment.

The experiment was not timed and the participants could modify their diagram or start all over again as often as they wanted. The completed diagram was collected from the participants individually when the participants felt like they were done. The diagram was stored into the XML format using SketchUML's built-in Save function and later converted into JPEG format for easier analysis.

Each participant was asked to fill out a uniform post-test survey after they completed the individual task. The post-test survey contained questions regarding the subjective quality of SketchUML, ease of use, entertainment aspects and missing features.

The post-experiment data analysis was performed in two parts. The first parts regarded the qualitative answers from the post-test questionnaire. The second part focused on the quantitative performance of the participants, i.e. how well the interface allowed them to fulfill their individual tasks. Since the participants were explained in detail how to use the interface, and since the participants were not unfamiliar with UML, a failure to achieve a certain goal (for instance, a connector could not be drawn or an attribute field was added incorrectly) is a result of a usability issue with the interface, which we seek to uncover in this study. The quantitative analysis could therefore be done by simply measuring the number of mistakes of a participants solution (i.e. the amount of diagram components, like connectors, or operation labels, that weren't drawn correctly in the diagram) and comparing the minimum number of steps needed to create an ideal solution (i.e. the number of gestures, actions and click-events minimally necessary to complete the task). The mistakes were divided into three groups: mistakes made during class symbol manipulation (i.e. when a class symbol was to be created, moved, or deleted), mistakes made during label manipulation (i.e. when a class symbol label or attribute or operation label was to be created, modified or deleted) and mistakes made during connector manipulation (i.e. when a connector like aggregation or generalization was to be created, modified or deleted).

III. Results

In the first condition, i.e. the task that was given to the first group, 9 mistakes were made overall. It took a minimum of 21 steps to complete the task. In total, no mistakes were made by any participant regarding class manipulation. Only one participant failed to correctly manipulate a connector – this participant created an aggregation instead of a generalization. The majority of 8 mistakes were done during label manipulation, with similar likelihood in every participant. These errors were exclusively errors in assignment of correct labels (e.g. two labels “is” and “Automatic” instead of one label “isAutomatic”). Each student made an average of 1.5 mistakes during label manipulation.

In the second condition, i.e. the task that was given to the second group, a total of 7 steps was minimally necessary to complete the task. A total of 16 mistakes were made, which corresponds to 3.2 mistakes per participant. Only two students incorrectly manipulated a class symbol, and a total of 5 mistakes were made during label manipulation. During connector manipulation, a total of 9 mistakes were made.

The qualitative post-test survey shows that the majority of the students ranked the overall quality of SketchUML as “medium to high”, which corresponds to a score of 4 on the 5-point Likert scale that was employed in the survey questionnaire. Open questions regarding the participants' subjective opinion of the program (their likes and dislikes) and their ideas for improvement uniformly contained criticism on

SketchUML's mode of feedback. Students mainly complained about difficulties to correctly spell names of attributes and operations. Positive feedback comprised the ease of creating connectors and class symbols. All students independently reported a low frustration level (despite the fact that they were not all able to complete the task correctly) and a high factor of enjoyment.

IV. Discussion and Future Work

The purpose of the Usability Testing with SketchUML was twofold: Firstly, the quality and usability of the newly developed software and the recently added features or SketchUML were to be assessed. Secondly, this study aims at understanding the advantage of Tablet PCs in non-collaborative educational work and in which way this technology can help to facilitate learning concepts in Software Engineering, exemplified at the UML syntax.

The first group completed a task that was designed to exploit all features of SketchUML's Student Mode, in which students have the ability to sketch UML components and thereby creating UML diagrams naturally in a pen-and-paper resembling minimalistic interface. The overall performance of the participants in this condition was very good, as only a minimal amount of mistakes were made by each student in drawing a diagram. The majority of the errors made were during label manipulation. Labels are deleted by using a scratch-out gesture and created or corrected by handwriting into the specific area. The post-test survey showed that no participants reported trouble when interacting with SketchUML using gestures, but the majority of criticism was directed towards poor recognition of the handwriting. This is consistent with the fact that no errors were made during class symbol manipulation (as this is mainly done by interaction through gestures) and with the fact that all label manipulation mistakes were made when assigning correct content to the label. The post-test survey further shows that the poor performance of the handwriting recognition engine is the mere reason for increased frustration, as it took some participants multiple attempts to correctly create a label. Although this significantly affects the efficiency of interacting with SketchUML, it did not affect the effectiveness, as all participants in the first group were able to create correct diagrams, compared to the target solution. The fact that one participant made one error during connector manipulation is assumed be caused by a lack of expertise with the UML syntax, as the student correctly created a connector – it was just the wrong type. Connector manipulation can therefore be considered both effective and efficient to use, as no student reported difficulties with such. Overall speaking, these findings suggest that the features of the Student Mode are naturalistic, effective, efficient and enjoyable. Improving the poor handwriting recognition performance will reduce the remaining frustration during label manipulation.

The second group was asked to complete a task that exploited all features of the Teacher Mode that differ from the Student Mode. The performance of the participants in this condition was not as high as in the first condition, yet remarkable. As the Teacher Mode gives slightly different visual feedback to certain gestures than the Student Mode, participants seemed to be somewhat confused. When creating a class or a label, it will occur with a red background color, to symbolize that this component was created by a teacher. This however, was incorrectly interpreted by some students who made mistakes during class symbol and label manipulation. Both errors during class manipulation and three of the five errors during label manipulation occurred when a student intended to create a class or a label and was uncertain about the red background. As a result, the previously correctly created class or label was removed. The other two mistakes during label manipulation were errors due to poor handwriting recognition. Similar in the first condition, the students' handwriting was incorrectly recognized, therefore rendering wrong label names. The majority of mistakes were made during connector manipulation in the second condition. When a connector is created in Teacher Mode, it will appear in red, to symbolize that this connector was created by a teacher. In contrast, when a connector, that was created in Student Mode (therefore appearing in black) is deleted in Teacher Mode, this connector will appear in red, with a red cross in the middle, symbolizing that this connector was removed by a teacher. Just like the problems during class symbol and label manipulation in the second group, this seems to be a problem with the mapping and feedback of the system to user input. Although the participants correctly understood that the red ink color depicts changes made in Teacher Mode, the difference between teacher created connectors (plain red ink) and teacher deleted connectors (red ink with a cross in the middle) does not appear to be obvious. Exclusively all mistakes have been made due to confusion of these two different connector drawing styles.

The post-test questionnaire further shows that SketchUML as an ink-only non-hybrid, non-traditional interface is very capable of satisfying user needs. The ease and effectiveness of the participants' interaction with the software shows that simplistic paper-and-pen resembling interfaces suffice in providing a good interaction with the user. Participants generally did not miss the ability to use the mouse, in fact, preferred using the pen as a pointing device over the traditional pointing devices. Also, the use of a keyboard was not missed, except for when handwriting recognition was not performed correctly. As a suggestion, it was mentioned that a virtual keyboard can help to manually correct incorrectly recognized handwriting. These results are very promising. Combined with the idiom of learning software, ink-enabled software for Tablet PCs have a high and important value in computer-aided learning (as also described in Bull et al and Bergue et al, both 2004), also when exposing simplistic interfaces, as it can be seen from this study.

SketchUML also proves to be a competent tool to facilitate learning the UML syntax. On basis of the findings of Qiu (2007), the recognition accuracy of gestures could be improved by a great deal so that SketchUML can be considered a naturalistic, learning facilitating tool for UML diagrams.

SketchUML has not reached a state of completeness. This study aimed at adding further functionality to SketchUML and assessing its usability in a formal evaluation. Work still needs to be done to improve the recognition performance of handwriting. From Qiu's findings earlier in 2007, gesture recognition was already improved significantly, but SketchUML needs to be more reliable in terms of handwriting. Also, this study has shown that there is some confusion about feedback in the Teacher Mode. In order to disambiguate the meaning of certain complex symbols, especially with regard to connectors, future work has to find a way to implement extended feedback abilities and a better mapping between gestures and their representing symbols. One idea to accomplish this task would be to include a status bar that informs the user what the last gesture was that the system recognized. Alternatively, tool tips that appear on the screen when the cursor is hovering over a symbol and explain what the underlying symbol represents can help to disambiguate the meaning of similar symbols. For the near future, further development is planned to add more UML-related features (i.e. symbols of the UML syntax that are currently not supported in SketchUML) as well as non-UML-related features, like undo/redo functionality, or a layout algorithm that supports the user to design uncluttered diagrams.

V. Acknowledgement

This project is made possible by the generous support of Dr. Gary Klatsky, Dr. Douglas Lea (both at SUNY Oswego) and Dr. Lin Qiu (formerly at SUNY Oswego) and, in part, The Research Foundation of SUNY through the generous funding of the Hewlett-Packard Development Company, L. P., HP Technology for Teaching Grant Number U06TFH0037C.

VI. References

Berque, D., Bonebright, T., and Whitesell, M. (2004). Using pen-based computers across the computer science curriculum. *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education.* Norfolk, Virginia, USA, pp. 61-65.

Bull, S., Bridgefoot, L., Corlett, D., Kiddie, P., Marianczak, T., Mistry C., Sandle, N., Sharples, M., Williams, D. (2004). Interacitive Lagbook: the development of an application to enhance and facilitate collaborative working within groups in higher education. *A book of papers from MLEARN 2004.* pp. 39-43.

Fowler, M. (2003). *UML Distilled, Third Edition: A Brief Guide to the Standard Object Modeling Language.* Addison-Wesley, Pearson Education.

Heines, J. M., Liang, W. T. (2007). Combining, Storing, and Sharing Digital Ink. *Proceedings of SIGCSE 2007.* Covington, Kentucky, USA.

Qiu, L. (2007). SketchUML: The Design of a Sketch-based Tool for UML Class Diagrams. *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications.* Chesapeake, Virginia, USA, pp. 986- 994.

Simon, B., Anderson, R., Hoyer, C., and Su, J. (2004). Preliminary experiences with a tablet PC based system to support active learning in computer science courses. *SIGCSE Bull.* Vol. 36, No. 3, pp. 213-217.