Towards a Framework for Supporting Software Modeling Activities Through Novel Interaction and Visualization Techniques

Mathias Frisch, Raimund Dachselt (advisor)
Computational Visualistics / Software Engineering Group
Otto-von-Guericke-University Magdeburg, Germany
{mfrisch, dachselt}@isg.cs.uni-magdeburg.de

Abstract

Though there is a multitude of software modeling tools available, the handling of diagrams, which are an essential way of representation, is still difficult. To overcome these problems this PhD thesis investigates the usage of novel visualization and interaction techniques for the software development process, including multi-touch displays, the interaction through zoomable user interfaces and the integration of diagrams drawn by hand. The goal of the work is to create a technical framework which offers respective techniques for the creation of and navigation in software models.

1. Motivation

Creating and interacting with diagrams is one of the key aspects of software engineering. On the one hand diagrams can represent very formal, complex and detailed models which are created for documentation purposes or serve for model driven development. As the unified modeling language (UML) is the most commonly used notation to design software systems, respective UML modeling tools are used to create and edit these diagrams. On the other hand diagrams can exist as hand drawings with an informal and transient character. They are node-link structures on paper or whiteboards created in different situations such as ad-hoc meetings. Various studies have been conducted (e.g., [3], [6]) to understand typical activities and needs of people who are involved in software development processes and to develop techniques which support the understanding and handling of diagrams. Some main requirements for tool support determined by these studies are:

- Giving a complete overview of the system and the visualization of dependencies between components.
- Providing different views for different people to the same content, e.g., a high level view for customers or managers and a low level system view for developers.
- Digitalization of handwritten diagrams and their integration in models or documents should be quick and easy.
- Collaborative work should be supported.
- Functions such as searching and filtering should be provided.

Current software modeling tools do not offer sufficient support for those needs. They implement logical interrelationships and dependencies but do not visualize them. Navigation in diagrams is often too cumbersome and a simultaneous view on macroscopic and microscopic levels of detail or a smooth transition between them is not possible. Besides that, there are devices such as Tablet PCs and electronic whiteboards which focus on the sketching aspect and try to combine the advantages of hand drawing with the ones of the digital world. They offer features such as infinite workspaces and the automatic conversion of sketches to formal models. But the problem of these devices is that they lack the flexibility of paper and often obstruct spontaneous sketching, as they are not always available and need power and time to start up.

The PhD project described in this proposal tries to solve these problems by applying and adopting novel visualization and interaction techniques to the domain of software development. The goal is to ease the creation and handling of software models in general and UML diagrams in particular, whereby interactive surfaces such as tabletop or wall displays are playing a central role. These devices can be used in both single and multi user scenarios. In single user scenarios touch and pen enabled tablets could be used to create and edit software models. In multi-user scenarios such as design meetings, wall displays and tabletops can serve as digital canvases for discussing, editing and merging diagrams. More precisely there are the following research hypotheses:

- The creation and navigation of visual models can be improved, with regard to speed and precision, by means of (multi) touch gestures and pen input.
- The comprehension of huge software models can be improved through zoombale user interfaces (ZUIs) and focus and context techniques which offer semantic zooming approaches and visual navigation aids.
- Techniques to transfer handwritten content to digital representations in a seamless way are beneficial especially in early phases of the software design process. This can be realized by means of digital pen and paper technologies. Users can work like with common paper, with the advantages of digital features available at the same time. As it is also possible to combine this approach with interactive surfaces, seamless interaction between paper and displays can be realized.

The overall goal is to create a technical framework which offers the interaction modalities described above and serves as a platform for user studies. It shall be used to compare the new techniques with state-of-the-art tools and to measure differences such as speed and accuracy between different approaches. Fig.1 summarizes the mentioned research directions and their relationships, thereby illustrating the conceptual structure of the framework. It will also serve as a guide for the research methodology of the PhD project.

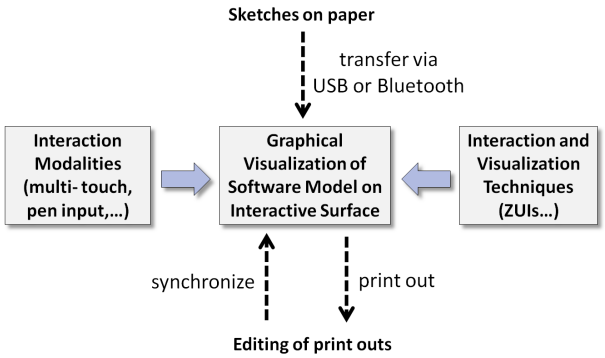

Figure 1. Conceptual structure of the framework.

The structure of this research proposal is as follows: section 2 deals with related work, details concerning the framework are presented in section 3 and section 4 discusses the results achieved so far and future work.

2. Related Work

Interactive Surfaces There is a multitude of projects which deal with multi-touch and hand gesture interaction on interactive surfaces such as tabletop or wall displays. However, the combination of touch and pen input has been hardly considered [1]. Furthermore, these kinds of devices have not yet been used for the visualization of and interaction with graph structures in general or software models in particular.

ZUIs In [12] a fisheye view technique on data flow diagrams and ER-diagrams is presented. It enables the user to see certain processes on a finer level of granularity and the content of clustered entities respectively. Beyond that, approaches such as [9] describe semantic zooming and focus and context techniques to navigate in package and class diagrams. A similar approach is implemented in ShriMP [11] which is an application for package structures and can also be used to explore ontologies and data flow diagrams. All the mentioned approaches try to solve the problem of limited overview and focus mainly on static diagrams (except for data flow diagrams). The interaction with behavior diagrams and the visualization of diagrams of different type nested into each other are not considered. An approach for the latter is presented in [10]. Here a special notation - not UML - is used to nest diagrams into each other. Besides that, this approach does not use semantic zooming with different LoDs and lacks intuitive and efficient interaction techniques.

Sketching Applications such as [8] and [2] were developed to support diagram-sketching during the software development process. They use devices such as Tablet PCs or electronic whiteboards and convert sketches in formal UML diagrams. However, they still lack the flexibility of common paper. Besides that, there are systems like [13] which try to integrate paper-based and digital interaction. They support the development of interactive paper applications through authoring and publishing functionalities. Though a multitude of applications were developed with these systems, digital pens and paper were neither applied to diagram drawing in general nor to UML in particular.

3. Interaction and Visualization Concepts

This section presents the features of the framework in more detail. Most of them are in conceptual stage, some are partly implemented.

3.1. Interaction Modalities

As mentioned in the introduction the focus is on the usage of interactive surfaces such as digital tablets, tabletop or wall displays to improve the creation of diagrams concerning speed and intuitiveness. Concerning interaction modalities for these devices, one key feature is the support of multi-touch in combination with pen-input. Thereby, finger gestures can be used to switch between different working

modes, to place diagram elements in the work space and to rearrange parts of a diagram. These ways of interaction can also take place with pens, whereas a pen can also be used to label elements or to create annotations by handwriting. Beyond that, finger gestures can be applied in conjunction with ZUIs. For example, a zooming process can be performed on a particular element by a certain gesture or parts of a diagram can be zoomed in after the respective elements have been selected by circling. More details concerning ZUIs are described in the following section. The touch and pen-interaction modalities will be integrated into the framework which should also be open to other interaction techniques.

3.2. Navigation and Interaction Techniques

Different visualization techniques are conceivable which display dependencies between model elements and allow a quick navigation between overview and detailed view. To accomplish the later the application of ZUIs with associated navigation aids to the software modeling domain are closer investigated. First results were presented in [7]. In particular semantic zooming can be applied to UML diagrams. This approach offers the possibility that elements of the respective diagram are displayed at different LoDs while zooming in and out. The amount and kind of information visible on a LoD varies depending on the available space and the semantic context, providing an overview with important parts still visible. This technique can be applied to a multitude of diagram types; for example on a coarse overview level of class diagrams classes are displayed as simple rectangles and just their labels and relationships are shown. By zooming in, classes are expanded and other elements such as attributes or cardinalities appear.

Besides giving a more comprehensive overview, semantic zooming can also be used for the visualization of refinements of diagrams by other diagrams, as refinements can be seen as nesting of diagrams into each other. Fig.2 illustrates this by means of a use case diagram whereby one of the use cases is refined by an activity diagram. By zooming in on a particular use case, a preview of nested elements is given (b). Users can choose between a nested activity diagram and a textual description, both describing the use case in more detail. The result of zooming on a nested activity diagram is shown in Fig.2 (c). There are additional navigation aids conceivable such as colored backgrounds and links (see Fig.2). They provide an indication to the user on which level of the hierarchy she is or if the current diagram is nested into another one. Further techniques which allow a quick navigation to off screen elements when zoomed in, focus and context techniques and multi-focus visualizations for filtering specific elements can be found in [7].

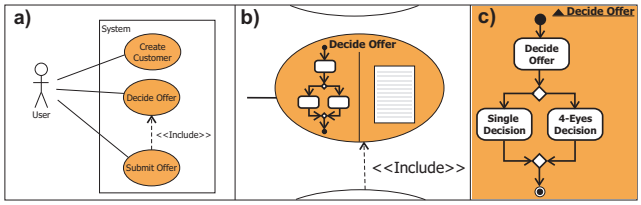

Figure 2. LoDs of a use case diagram. "Decide Offer" is further refined by a detailed text description and an activity diagram.

3.3. Sketching with Digital Pens & Paper

Commonly, digital pens and paper technologies use the Anoto functionality. Equipped with an integrated image processor and an infrared camera, the pens take snapshots of the Anoto dot pattern which is printed almost invisibly on the paper. Every snapshot delivers sufficient information to determine the absolute position of the pen on paper and what the user has written or drawn. Data can be transferred from the pen to an application by putting the pen in a connected docking station or by streaming immediately via Bluetooth.

As presented in [5] this technique can also be applied to sketched diagrams which are recognized and converted to formal representations. To support recognition UML sketch books are conceivable. They offer boxes at the bottom of each page representing different diagram types (see Fig.3 left). By tapping for example on the "class diagram" box, the application tries to convert as many parts of the transferred sketch as possible into class diagram elements. Besides that, connections between elements can be drawn across different sheets of paper. This can be beneficial for collaborative settings and solve the problem of lacking space. Digital representations can also be printed out on Anoto-enabled paper, the printed diagrams can be edited and synchronized again with the digital version.

Furthermore, in our lab we use the Anoto functionality for multi-pen enabled displays. The pattern is located underneath its surface; thus it is possible to draw like on whiteboards and to investigate seamless transitions of sketches from the paper to the display. This can be beneficial in meeting scenarios, for example when a diagram which a developer has sketched by himself should be discussed in a group or combined with solutions of other team members. Another way of interaction can happen by means of paper palettes, which software designers can hold in their hands. On each palette, e.g. UML elements are printed and by tapping with the pen the respective element is "picked up" and can be placed on the interactive surface. In that way it is possible to create ad hoc formal UML diagrams, especially

in collaborative work settings (see Fig.3 right).

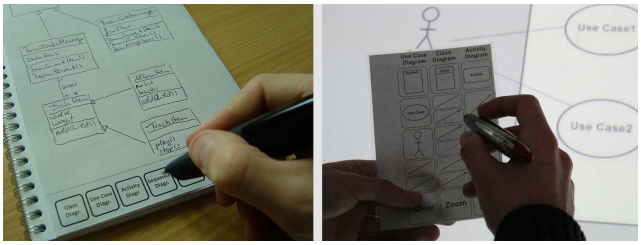

Figure 3. UML Sketchbook (left) and paper UML-palette (right).

4. Status of the Work

Results achieved to date As a start, two prototypes for digital pens and paper were implemented. One realizes the aforementioned UML palettes and enables multiple users to tap on UML elements arranged on a laminated paper palette. The elements selected in that way can be placed on an pen-enabled interactive surface (see Fig.3 right). The second prototype implements a sketch recognition application which can recognize rectangles, ellipses and connections (see Fig.4). The prototype can be used with Anoto-enabled paper as well as with tabletops.

Besides that, a qualitative study is currently conducted which questions software modeling activities in companies and the role of software models in communication between IT and domain experts. It expected to reveal needs people have who are involved in software development processes, when they create and interact with diagrams (both formal and sketched ones). This study shall underpin requirements arising from literature and serve as a base to find the most promising approaches presented in the aforementioned sections.

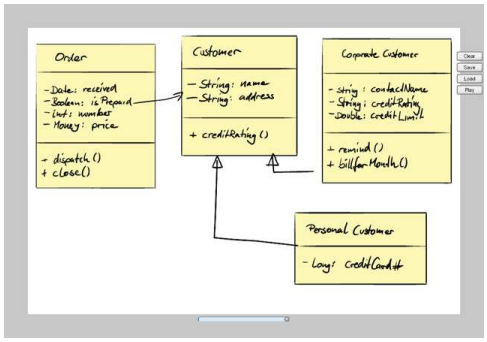

Figure 4. The sketch recognition prototype.

Future work The current main focus is on the development of a set of multi touch gestures for the interaction with software diagrams. The system shall also offer the semantic zooming approach for which several sketches and animations were created to analyze different ways of interaction and visualization [4]. The next steps will include the integration of the UML-palette application and the sketch recognition prototype. Beyond that we plan to conduct user studies to compare state-of-the-art tools with these novel approaches, concerning speed, accuracy and user experience. Furthermore, it is important to evaluate the combination of the proposed techniques with existing workflows if possible in field studies.

References

[1] P. Brandl, C. Forlines, D. Wigdor, M. Haller, and C. Shen. Combining and measuring the benefits of bimanual pen and direct-touch interaction on horizontal interfaces. In *Proc. of AVI '08*, pages 154–161, 2008.

[2] Q. Chen, J. Grundy, and J. Hosking. An e-whiteboard application to support early design-stage sketching of uml diagrams. In *Proc. of HCC '03*, pages 219–226, 2003.

[3] M. Cherubini, G. Venolia, R. DeLine, and A. J. Ko. Let's go to the whiteboard: how and why software developers use drawings. In *Proc. of CHI '07*, pages 557–566, 2007.

[4] University of Magdeburg, Germany, supplemental Material for this paper, http://isgwww.cs.uni-magdeburg.de/cvse/Forschung/SZofUML/.

[5] R. Dachselt, M. Frisch, and E. Decker. Enhancing uml sketch tools with digital pens and paper. In *Proc. of SoftVis '08*, pages 203–204, 2008.

[6] U. Dekel. Supporting distributed software design meetings: what can we learn from co-located meetings? In *: Proc. of HSSE '05 Workshop*, pages 1–7, 2005.

[7] M. Frisch, R. Dachselt, and T. Brückmann. Towards seamless semantic zooming techniques for uml diagrams. In *Proc. of SoftVis '08*, pages 207–208, 2008.

[8] T. Hammond and R. Davis. Tahuti: A geometrical sketch recognition system for uml class diagrams. *Papers from the AAAI Spring Symposium on Sketch Understanding*, pages 59–68, 2002.

[9] T. Jacobs and B. Musial. Interactive visual debugging with uml. In *Proc. of SoftVis '03*, pages 115–122, 2003.

[10] T. Reinhard, S. Meier, R. Stoiber, C. Cramer, and M. Glinz. Tool support for the navigation in graphical models. In *Proc. of ICSE '08*, pages 823–826, 2008.

[11] M.-A. Storey, C. Best, J. Michaud, D. Rayside, M. Litoiu, and M. Musen. Shrimp views: an interactive environment for information visualization and navigation. In *CHI '02: extended abstracts on Human factors in computing systems*, pages 520–521, 2002.

[12] O. Turetken, D. Schuff, R. Sharda, and T. T. Ow. Supporting systems analysis and design through fisheye views. *Commun. ACM*, 47(9):72–77, 2004.

[13] R. B. Yeh, A. Paepcke, and S. R. Klemmer. Iterative design and evaluation of an event architecture for pen-and-paper interfaces. In *Proc. of UIST '08*, pages 111–120, 2008.