FlexiView Experimental Tool: Fair and Detailed Usability Tests for Requirements Modeling Tools

Parisa Ghazi, Martin Glinz
Department of Informatics, University of Zurich, Switzerland
{ghazi, glinz}@ifi.uzh.ch

***Abstract*—Enhancing the usability of tools such as requirements modeling tools requires several cycles of testing and improvement. Since this process is costly, it is usually ignored.**

In this paper, we present an experimental tool which we have developed with two goals: (i) comparing the usability of a new navigation technique for requirements artifacts called FlexiView with traditional zooming and scrolling, and (ii) developing a platform that enables fast implementation and fair usability comparisons of new navigation techniques while producing generalizable results.

Demo video: https://youtu.be/rf_dPvwB5hY

***Index Terms*—Graphical Models, Requirements Engineering, Modeling Tools, User Interface**

I. Introduction

Requirements engineers elicit various types of information, analyze them and store them in artifacts. In order to conveniently read, understand, modify and share the information in these artifacts, they choose suitable formats such as documents, charts, and diagrams [1], and use requirements modeling tools (RMTs) for working with these artifacts. Thus, RMTs greatly influence the performance of requirements engineers.

In addition to the suitability of the functional features of a tool, its usability also impacts the performance of doing a task. In the definition of usability by the ISO 9241-11 standard [2], three properties of using a product are considered: efficiency, effectiveness, and satisfaction. To improve the usability of a tool, several cycles of testing and improvement are required. In each cycle, (i) new solutions for the known challenges are implemented, (ii) usability tests are carried out and (iii) the solutions are updated based on the results of the tests.

Different aspects of an RMT influence the usability factors. We focus on the following aspects: (i) how the information is presented on the screen and (ii) how the users can customize this presentation to fit their needs at a certain moment. The process of usability testing and improvement is costly and thus mainly ignored when developing RMTs. As a result, the graphical user interfaces of RMTs have not changed significantly since their emergence in the 1980s, although users experience major challenges especially when working with interconnected artifacts that are larger than the screen [3].

In this paper, we present the tool that we developed for performing usability tests on FlexiView [4]. FlexiView is a new physics-based focus+context navigation technique for large and interconnected requirements artifacts. In FlexiView, based on the current task, the user indicates interesting elements of a graphical model by putting virtual magnets on them. Based on the strength of the magnets, FlexiView allocates screen space to the elements. The result is a view of the diagram with all elements, in which the more interesting elements are larger and reveal more details than the other elements.

Our goal was twofold: (i) creating a platform for evaluating FlexiView by comparing it to traditional navigation techniques and (ii) creating a framework for enhancing the general process of usability testing and improvement cycles for RMTs by making usability testing cheaper and faster. In order to decrease the cost of usability test cycles, a framework is needed that allows *fast* implementation and modification of solutions. Meanwhile, it should provide *fair* comparisons with *generalizable* results.

For having fairness in comparisons, the solutions that are being compared should be tested in equal situations. Implementing the solutions in different tools with different features does not guarantee that the observed differences are only due to the differences between the solutions. Therefore, both solutions should be either implemented in an existing RMT or in an experimental tool. In the first case, since an existing tool is already complicated, integrating a new solution into it and modifying it after a round of usability tests is time-consuming. In the second case, while implementing solutions is faster in an experimental tool, the test cases are limited due to the lack of advanced features, which reduces generalizability.

We decided to implement zooming, scrolling, and FlexiView navigation in an experimental tool for having *fast* implementations and *fair* comparisons. We used ImitGraphs instead of requirements models to preserve the *generalizability* of the results. ImitGraphs are versions of simple node-and-edge graphs with additional parameters that allow them to imitate the behavior of other models [5].

II. The FlexiView Experimental Tool

In essence, our tool is an ImitGraphs modeling tool for medium-sized tablets with zooming, scrolling, and FlexiView navigation techniques. It can log user interactions with high detail and allows the usability tester to define tasks. We call it *experimental* as it permits easy and fast experimentation.

Our tool has two types of users: usability testers and users of the usability tests. The usability testers have three major use cases: (i) customizing ImitGraphs by defining different types of nodes, connections and joints, (ii) creating different usability testing tasks by defining the input files, the time limit

2332-6441/18/$31.00 ©2018 IEEE
DOI 10.1109/RE.2018.00069

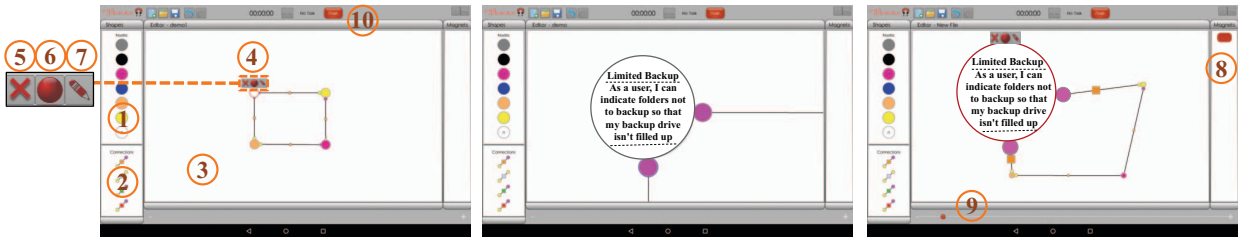

Fig. 1. Three screenshots of the tool while showing a simple diagram (a) and while the upper-left node is zoomed in (b) and magnetized (c).

and which navigation techniques are allowed in each task, (iii) loading tasks and starting them during the usability tests.

The users of the usability tests start after the usability tester loads a task. Figure 1 shows three screenshots of the tool with highlighted numbers (1-10). The left-side menu contains the nodes (1) and connections (2) that are defined by the testers. In the middle of the screen, the editor area is located (3). The users can create new nodes and connections by dragging them from the side menu to the editor area. Within the editor area, the users can move nodes, joints, and connections by one-finger dragging. If a node and a joint of a connection are allowed to be attached in the ImitGraph definition, the user can attach them by moving the joint close to the node. When tapping on nodes and connections, a context menu appears above the selected object (4). The users can remove an object and edit its text by pushing the corresponding buttons (5 and 7 respectively) in the context menu.

In Figure 1a, the editor area contains an ImitGraph composed of four nodes and four connections. In Figure 1b, the user zoomed in on the upper-left node in order to see its details. Zooming and scrolling are possible by a pinch gesture.

In the FlexiView navigation, three parts of the user interface are involved: context menu, magnet pane (8) and magnet slider (9). In Figure 1c the user magnetizes the upper-left node to see its details. To achieve this, first, a magnet is put on a node by pushing the corresponding button in the context menu (6). Then, via the slider at the bottom (9), the strength of the magnet can be adjusted. As soon as a magnet is created, a red button appears in the magnet pane (8) which represents that magnet. When a user clicks a magnet button in the magnet pane, the node in the model which carries the corresponding magnet is selected. Thus, the magnet pane provides an overview of all currently set magnets.

During the usability test trials, the tool automatically collects experiment data by recording every interaction of the users with millisecond precision. The recordings comprise two types of interactions: instant and continuous. Instant interactions do not have a duration, e.g., selecting a node. The continuous interactions take place over a period of time which can be less than a second or up to several seconds, e.g., moving a node. The tool records one entry for each instant interaction and multiple entries for continuous interactions depending on their duration. The task menu at the top (10) shows a timer and a finish button. The timer encourages the users to perform the task as fast as they can.

Different types of information can be extracted from the raw data recorded by our tool. Let us consider this scenario: *The user is given a diagram with a hierarchical structure and is asked to search the hierarchy to find the nodes with a certain keyword using zooming and scrolling*. In this scenario, the user has to zoom in to check the details of each node. When zoomed in, the connections of the nodes are not recognizable. Therefore, after checking a node, the user needs to zoom out to view the overall structure of the diagram and plan for visiting the next nodes. The user continuously zooms in and out until all nodes are checked. From the recorded data we can find out how much time the user spent on checking, planning and navigating. Additionally, we can visualize the users' search path and count the nodes that they missed or checked more than once. A similar scenario can be devised for FlexiView which creates the opportunity of comparing these techniques.

III. Conclusion and Future Work

We have conducted first experiments with the FlexiView tool and found that it is capable of gathering valuable user interaction data with high accuracy and detail [6]. This proved to be helpful in comparing the usability of FlexiView in comparison with classic zooming and scrolling. Using ImitGraphs speeded up the process of creating our experimental tool while it allowed us to experiment with ImitGraphs that imitated the behavior of requirements engineering models such as activity diagrams, class diagrams, and goal decomposition models. Finding the exact extent to which the results are generalizable needs further research. Next, we want to use our tool for conducting comparisons of other navigation techniques.

References

[1] J. M. C. De Gea, J. Nicolás, J. L. F. Alemán, A. Toval, C. Ebert, and A. Vizcaíno, "Requirements engineering tools: Capabilities, survey and assessment," *Info. Soft. Tech., 54, 10*, pp. 1142–1157, 2012.

[2] Intl. Std. Org. (ISO), *ISO 9241-11:2018: Ergonomics of human-system interaction – Part 11: Usability: Definitions and concepts*, 2018.

[3] P. Ghazi and M. Glinz, "Challenges of working with artifacts in requirements engineering and software engineering," *Req. Eng. (REJ),22,3*, pp. 359–385, 2017.

[4] P. Ghazi, N. Seyff, and M. Glinz, "FlexiView: A magnet-based approach for visualizing requirements artifacts," in *21st Int. Conf. on Req. Eng. Foundation for Soft. Quality (REFSQ'15)*, pp. 262–269.

[5] P. Ghazi and M. Glinz, "ImitGraphs: Towards faster usability tests of graphical model manipulation techniques," in *9th Int. Workshop on Modeling in Soft. Eng. (MiSE@ICSE2017)*, pp. 61–67.

[6] ——, "An experimental comparison of two navigation techniques for requirements modeling tools," in *26th IEEE Int. Req. Eng. Conf. (RE'18)*.