Flexisketch team: Collaborative Sketching and Notation Creation on the Fly

Wüest, Dustin ; Seyff, Norbert ; Glinz, Martin

Abstract: When software engineers collaborate, they frequently use whiteboards or paper for sketching diagrams. This is fast and flexible, but the resulting diagrams cannot be interpreted by software modeling tools. We present FLEXISKETCH TEAM, a tool solution consisting of a significantly extended version of our previous, single-user FLEXISKETCH tool for Android devices and a new desktop tool. Our solution for collaborative, model- based sketching of free-form diagrams allows users to define and re-use diagramming notations on the fly. Several users can work simultaneously on the same model sketch with multiple tablets. The desktop tool provides a shared view of the drawing canvas which can be projected onto an electronic whiteboard. Preliminary results from an exploratory study show that our tool motivates meeting participants to actively take part in sketching as well as defining ad-hoc notations. Demo video: http://youtu.be/0kHjNfHLViM

DOI: https://doi.org/10.1109/ICSE.2015.223

2015 IEEE/ACM 37th IEEE International Conference on Software Engineering

FLEXISKETCH TEAM: Collaborative Sketching and Notation Creation on the Fly

Dustin Wüest Norbert Seyff Martin Glinz
Department of Informatics, University of Zurich, Switzerland
{wueest, seyff, glinz}@ifi.uzh.ch

***Abstract*—When software engineers collaborate, they frequently use whiteboards or paper for sketching diagrams. This is fast and flexible, but the resulting diagrams cannot be interpreted by software modeling tools. We present FLEXISKETCH TEAM, a tool solution consisting of a significantly extended version of our previous, single-user FLEXISKETCH tool for Android devices and a new desktop tool. Our solution for collaborative, model-based sketching of free-form diagrams allows users to define and re-use diagramming notations on the fly. Several users can work simultaneously on the same model sketch with multiple tablets. The desktop tool provides a shared view of the drawing canvas which can be projected onto an electronic whiteboard. Preliminary results from an exploratory study show that our tool motivates meeting participants to actively take part in sketching as well as defining ad-hoc notations.**

Demo video: http://youtu.be/0kHjNfHLViM

I. INTRODUCTION

Software engineers frequently use whiteboards when they collaborate with each other or with project stakeholders in early project meetings. They create diagrammatic sketches for requirements elicitation, solution and problem design, viewpoint negotiation, and idea generation [1], [2]. The resulting sketches often show notations that deliberately deviate from defined standards such as UML, are simple, and sometimes ambiguous [2], [3]. On the one hand, creating such notations ad-hoc allows meeting participants to depict problems and ideas at any level of detail and in a form that can be understood by all participants. On the other hand, using non-standard (and potentially ambiguous) notations makes sketches hard to understand outside of the meeting context [4]. People who did not attend a meeting have to assume meanings for symbols. Even meeting participants might no longer be able to correctly interpret the sketches a few weeks later [1], [5]. For re-using sketches during the software engineering process, engineers tend to either include pictures of them in documents, or manually build formal models from scratch, based on the sketches. The latter can be a time-intensive and error-prone task [5].

In our previous work, we developed FLEXISKETCH [5], [6], a tablet-based tool for free-form sketching and the creation of arbitrary node-and-edge diagrams. FLEXISKETCH provides lightweight metamodeling functionality that allows the user to step-wise formalize the diagram sketches by assigning types and cardinality rules to sketched elements.

The main idea of our tool, which distinguishes it from other sketching tools, is that users can freely and seamlessly interleave sketching and metamodeling (i.e., defining a syntax for sketched symbols and links). This gives users the opportunity of both unconstrained sketching and modeling with a defined notation (that users may create themselves), including any combination of the two options.

So far, FLEXISKETCH has been an application for a single user. In this paper, we present FLEXISKETCH TEAM, a tool that supports synchronous, co-located, and multi-display collaboration for sketching and modeling in meetings. It consists of a significantly extended tool version for tablets and the new FLEXISKETCH DESKTOP for PCs that provides similar functionality and acts as a server. Our approach allows multiple users to work on the same sketch concurrently. Using their own tablets, participants can collaboratively work on a problem and also define a modeling notation ad hoc. This notation can be re-used in later software design sessions. The screen of the PC tool can be projected onto a wall or an electronic whiteboard and provides a shared view of the sketch canvas and the defined language constructs.

In the remainder of the paper, we first describe the core features of FLEXISKETCH (Sect. II) and then present the new tool solution for collaborative work (Sect. III), followed by preliminary evaluation results (Sect. IV). Sect. V discusses related work and Sect. VI concludes.

II. FLEXISKETCH BASIC

The single-user version of FLEXISKETCH is an Android tool[1] for model-based sketching (see Figure 1) [5]. Users can start by drawing free-form sketches. Whenever they lift the finger for a specified amount of time, the strokes are converted into a distinct symbol. Symbols can be selected and moved around. A context menu allows for further manipulation (e.g., resizing, adding text, deleting). A new feature allows users to import existing images into a sketch. Images behave like symbols and can be cropped. A link between two symbols is created by drawing a stroke from one symbol to another. Links have a context menu similar to symbols. The context menu enables users to add semantics to symbols and links by assigning types and cardinality rules, and thereby define the vocabulary of a modeling language. All typed elements are copied into a type library. The type library is our construct that holds all types and the visual notation of a modeling language. A sketch recognition algorithm recognizes symbols

[1] An early version of our tool is available in the Google Play store. A video demonstrating its features can be found at http://youtu.be/D06t0K5Otzw

978-1-4799-1934-5/15 $31.00 © 2015 IEEE
DOI 10.1109/ICSE.2015.223

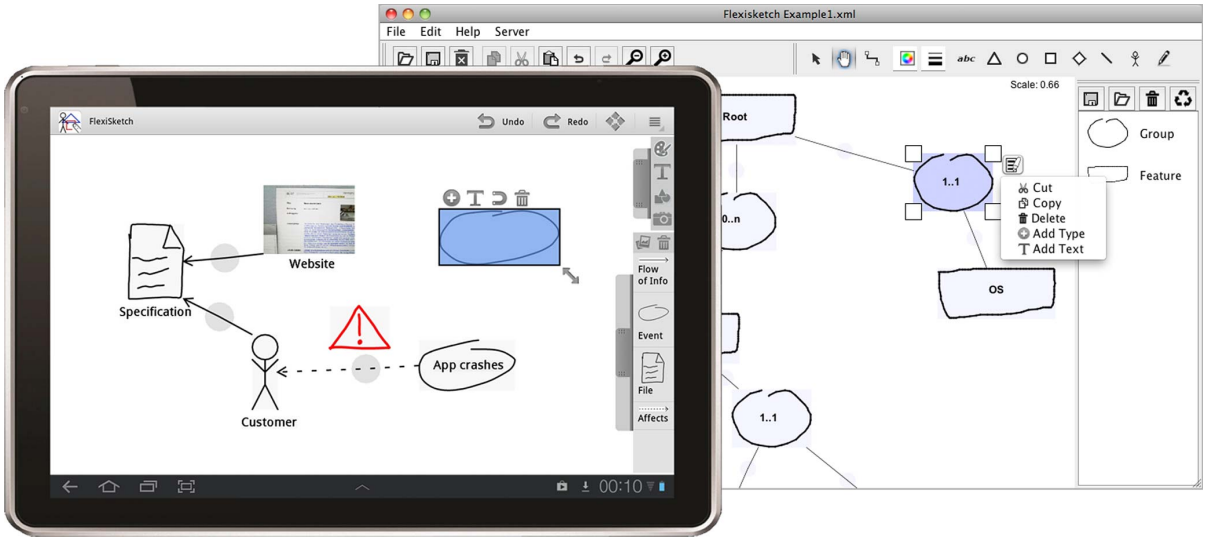

Fig. 1. The Android version of FlexiSketch running on a tablet, and the desktop version running on a Mac (showing two different model sketches).

that resemble user-defined types. By dragging a slider on the right side of the screen, users can reveal a container showing the entries of the type library. A drag&drop mechanism lets users re-use elements from the library. Type libraries can be changed at any time during the sketching process. The tool automatically builds a metamodel and infers cardinality rules for links, according to the current sketch and the definitions in the library. Both the sketches and the metamodel can be exported in a structured form (e.g., xml files, see Figure 2). We provide more details about our tool's metamodeling capabilities and a step-wise formalization of model sketches in [6].

III. FLEXISKETCH TEAM

The envisaged usage scenarios for our tool are requirements, design, and idea generation meetings in software development (see Section 1). We currently focus on co-located settings where communication between participants, apart from sketching, happens via natural language and gestures. Supporting geographically distributed synchronous collaboration is also possible, but not yet implemented.

Our tool provides a multi-screen setup where all meeting participants run the app on their tablets and have concurrent editing access to a synchronized canvas. The interface of FLEXISKETCH DESKTOP looks slightly different since it is adjusted to mouse & keyboard input (see Figure 1). It is also suited for electronic whiteboards: all actions can be performed by using the left mouse button (which is simulated by a touch), and the various parts of the interface (drawing palette, type library, etc.) can be moved and placed anywhere on the screen.

When participants meet, they connect their tablets to a server (a computer running FLEXISKETCH DESKTOP) via an ad-hoc wifi network, by e.g., using one of the Android tablets as mobile hotspot. Each participant has the option to scroll and zoom his/her own view to focus on different parts of the canvas. Optionally, the server is connected to a projector that

```
...
<Symbol>
        <type> File </type>
        <attributes>
                <labels> ... </labels>
                <min__occurrence> 0 </min__occurrence>
                <max__occurrence> N </max__occurrence>
        </attributes>
</Symbol>
<Link>
        <type> Flow of Info </type>
        <appearance> ... </appearance>
        <direction> UNIDIRECTIONAL </direction>
        <connections>
                <connection__1>
                        <from__element> Person </from__element>
                        <to__element> File </to__element>
                        <from_cardinalities>
                                <min> 0 </min> <max> 1 </max>
                        </from_cardinalities>
                        <to_cardinalities>
                                <min> 0 </min> <max> 1 </max>
                        </to_cardinalities>
                </connection__1>
        </connections>
</Link>
...
```

Fig. 2. Metamodel excerpt from the left sketch in Figure 1.

provides a high-resolution overview of the sketch canvas and a list of all defined elements. This overview is automatically zoomed such that it always shows all drawings, but it can also be zoomed manually to focus on certain parts of the canvas and steer discussions (see Figure 3).

Drawings are synchronized as soon as strokes from a user get converted into a distinct element. The Android app adds a unique element and user ID, and sends the element to the server, which forwards it to all other tablets. For all subsequent manipulations of existing elements, only the IDs and the actual change (the delta) are communicated over the network. This helps to keep network traffic low and minimizes the time to propagate changes. The same applies to changes in the type library. The notation is synchronized between tablets and the

Fig. 3. Meeting participants collaboratively create and discuss a model. The workspace is synchronized across the tablets and the electronic whiteboard.

Fig. 4. A symbol on the left tablet is selected and appears in blue. On all other tablets, the symbol appears in red and is locked.

libraries get immediately updated whenever a user assigns a type to an element or deletes a type from the library. The fact that sketches and notations are synchronized across tablets implies that all participants have the meeting results on their personal tablets when they leave the meeting.

Users can not only sketch simultaneously within the same region of the canvas, but also define types of different elements concurrently. If two elements are created and the same type is assigned to both of them, the tool only generates one type entry, but stores both elements as alternative representations for that type. In contrast, when two users try to manipulate the same element at the same time, a non-optimistic locking mechanism prevents this. Only one user can access the context menu of an element at a time (to perform actions such as move, scale, delete, add text, assign a type, or define cardinalities). The main reason for the locking mechanism is to prevent inconsistent states of individual elements. Otherwise a user might, e.g., delete an element while another user is looking at the text entry popup for adding text to this element. For the manipulation of an element, the user first selects the element by tapping on it. The server locks the element on all other tablets where it no longer reacts to inputs and is shown with a red background (Figure 4). On the user's tablet, the element appears selected and shows its context menu. All manipulations performed by the user are immediately propagated. The element gets unlocked when the user deselects it (by successfully performing a manipulation or tapping the white part of the canvas). The locking mechanism also provides some user awareness in the form of visual cues showing what parts of the model are currently manipulated by other users.

FLEXISKETCH TEAM includes a share function that allows any user to push his/her current sketch and notation to the other users' tablets. This function can be used in a scenario where workshop participants prepare model sketches before the actual meeting, or when they individually sketch during the meeting. Participants can join an ongoing meeting and receive the current meeting artifacts with the push of a button. Furthermore, a user can disconnect her tablet to have a private workspace. When she is ready, she can re-connect and share her work with the other users.

IV. PRELIMINARY EVALUATION RESULTS

We conducted a qualitative study where we video recorded simulated workshops with three student teams and three SE practitioner teams from industry. Each team consisted of three co-located members. Here we concentrate on preliminary results from analyzing the practitioner teams. The teams were asked to choose a current SE related task or problem from their organization as a collaborative ideation and modeling task. We investigated how they collaboratively sketched and defined notations on the fly when supported by our tool.

We found that all participants actively took part in the sessions and used the possibility to sketch simultaneously. In each team, the notations were defined by multiple participants. This happened incrementally during the sketching task, whenever they introduced a new element type, i.e., they interleaved sketching and metamodeling activities. All groups have chosen notations loosely based on existing standards by first agreeing on a known diagram type and then deliberately deviating from the notation standards (see Figure 5). Hence, discussions about semantics happened during the whole workshops. The type library with its drag&drop mechanism was heavily used. The possibility to re-use types motivated participants to define them, and led to diagrams with consistent notations. Regarding the usefulness of our tool, participants stated that defining types can also be seen as a form of documentation which helps to convey the meaning of elements and the sketch as a whole for later re-use. The practitioner groups reported that they enjoyed the flexibility of our tool compared to other software modeling tools, and that it is faster to share the created artifacts with others compared to classic whiteboards. They stated that they would prefer whiteboards for short-lived, small sketches, while they favor FLEXISKETCH for larger sketches, as well as for sketches that are, or will be, re-used.

V. RELATED WORK

Collaborative sketching is an important method to foster creativity and discuss design ideas [3], [7]. Studies show that teams deliberately deviate from standard notations during early design meetings, e.g., Dekel and Herbsleb [4], and Ossher et al. [8]. A tool can only support such ad-hoc notations if it provides some kind of metamodeling mechanisms. While it was long believed that metamodeling should only be done by experts – and it has been shown that end-user metamodeling is indeed hard to achieve (e.g., [9]) –, we argue that a form of lightweight metamodeling (or "just enough metamodeling") can be achieved in an end-user friendly way and is powerful enough for allowing the export of sketches as models.

The ability to collaboratively formalize arbitrary sketches of node-and-edge diagrams in a step-wise manner distinguishes FLEXISKETCH TEAM from similar work, which either provides sketch interfaces and recognition for predefined lan-

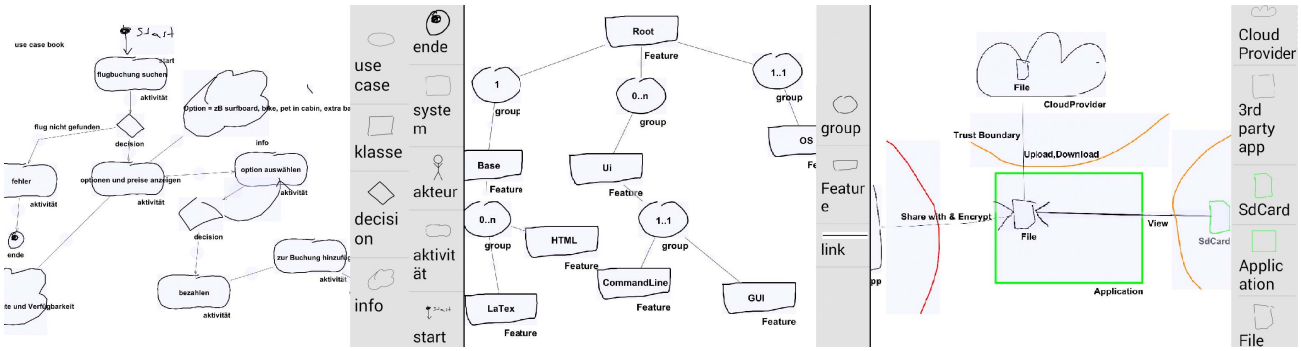

Fig. 5. Result extracts from practitioner teams. The grey boxes show defined elements. Here, types are also displayed to the bottom right of each element.

guages only, or allows for free-form sketching with little support for formalization. Examples for the former case include NetSketcher [10], and Scribble [11] which can inject sketching functionality into existing GEF based editors. CEL [12] is a mobile tool that uses a minimal set of predefined element types, and exports models as skeleton source code. Examples for the latter are IdeaVis [13] and TEAM STORM [14]. Calico [2] and Idea Playground [15] allow for grouping and node-and-edge structures in free-form sketches. Some tools such as Knight [16] and Tivoli [17] include both informal and formal drawing modes, but no formalization of ad-hoc notations.

The Electronic Cocktail Napkin [3] provides functionality similar to the single-user FLEXISKETCH tool, but is more complex and needs programming/scripting knowledge. BitKIT also includes an incremental formalization approach [18], but focuses on specific data structures such as tables, while we focus on node-and-edge structures. Furthermore, FLEXISKETCH TEAM can be used to create a first draft of a custom modeling language (i.e., a DSL) by people without metamodeling expertise. Other metamodeling tools are hard to understand for non-experts [19] and/or do not allow seamless switching between modeling and metamodeling.

VI. CONCLUSION

In this paper we presented our multi-screen, node-and-edge diagram sketching tool for software engineering. With FLEXISKETCH TEAM, participants can sketch simultaneously and use lightweight metamodeling mechanics to collaboratively define custom notations on the fly and step-wise formalize the drawings. The latter two features differentiate our approach from related work. Preliminary evaluation results indicate that our tool fosters interleaving of sketching and type-defining activities, and motivates all group members to take part in both activities. The groups managed to define consistent notations. In future work, we plan to extend our approach with user awareness and communication features in order to support distributed collaboration. We plan to evaluate our tool in real software projects. We will investigate how sketches made with our tool are re-used and changed during projects, and gather feedback about the quality of sketches from the people who will actually re-use these artifacts.

REFERENCES

[1] 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. CHI*, 2007, pp. 557–566.

[2] N. Mangano, T. D. LaToza, M. Petre, and A. van der Hoek, "Supporting informal design with interactive whiteboards," in *Proc. CHI*, 2014, pp. 331–340.

[3] M. D. Gross and E. Y.-L. Do, "Ambiguous intentions: a paper-like interface for creative design," in *Proc. UIST*, 1996, pp. 183–192.

[4] U. Dekel and J. D. Herbsleb, "Notation and representation in collaborative object-oriented design: An observational study," *SIGPLAN Not.*, vol. 42, no. 10, pp. 261–280, Oct. 2007.

[5] D. Wüest, N. Seyff, and M. Glinz, "FlexiSketch: A mobile sketching tool for software modeling," in *Proc. MobiCASE*, 2013, pp. 225–244.

[6] D. Wüest, N. Seyff, and M. Glinz, "Semi-automatic generation of metamodels from model sketches," in *Proc. ASE*, 2013, pp. 664–669.

[7] R. Van der Lugt, "Functions of sketching in design idea generation meetings," in *Proc. C&C*, 2002, pp. 72–79.

[8] H. Ossher, B. John, M. Desmond, and R. Bellamy, "Are flexible modeling tools applicable to software design discussions?" in *Workshop on Studying Professional Software Design*, vol. 12, 2010.

[9] J. Sánchez-Cuadrado, J. De Lara, and E. Guerra, "Bottom-up meta-modelling: an interactive approach," in *Proc. MODELS*, 2012, pp. 3–19.

[10] N. Baloian, G. Zurita, F. Santoro, R. Araujo, S. Wolfgan, D. Machado, and J. Pino, "A collaborative mobile approach for business process elicitation," in *Proc. CSCWD*, June 2011, pp. 473–480.

[11] A. Scharf and T. Amma, "Dynamic injection of sketching features into GEF based diagram editors," in *Proc. ICSE*, 2013, pp. 822–831.

[12] R. Lemma, M. Lanza, and F. Olivero, "CEL: modeling everywhere," in *Proc. ICSE*, 2013, pp. 1323–1326.

[13] F. Geyer, H.-C. Jetter, U. Pfeil, and H. Reiterer, "Collaborative sketching with distributed displays and multimodal interfaces," in *Proc. ITS*, 2010, pp. 259–260.

[14] J. Hailpern, E. Hinterbichler, C. Leppert, D. Cook, and B. P. Bailey, "TEAM STORM: Demonstrating an interaction model for working with multiple ideas during creative group work," in *Proc. C&C*, 2007, pp. 193–202.

[15] F. Perteneder, C. Grossauer, T. Seifried, J. Walny, J. Brosz, T. Tang, and S. Carpendale, "Idea Playground: When brainstorming is not enough," in *Designing Collaborative Interactive Spaces for e-Creativity, e-Science and e-Learning Workshop, AVI*, 2012.

[16] C. H. Damm, K. M. Hansen, and M. Thomsen, "Tool support for cooperative object-oriented design: Gesture based modelling on an electronic whiteboard," in *Proc. CHI*, 2000, pp. 518–525.

[17] E. R. Pedersen, K. McCall, T. P. Moran, and F. G. Halasz, "Tivoli: An electronic whiteboard for informal workgroup meetings," in *Proc. INTERACT and CHI*, 1993, pp. 391–398.

[18] N. Mangano, H. Ossher, I. Simmonds, M. Callery, M. Desmond, and S. Krasikov, "Blending freeform and managed information in tables (NIER Track)," in *Proc. ICSE*, 2011, pp. 840–843.

[19] J. Grundy, J. Hosking, K. Li, N. M. Ali, J. Huh, and R. L. Li, "Generating domain-specific visual language tools from abstract visual specifications," *IEEE Trans. Softw. Eng.*, vol. 39, no. 4, pp. 487–515, Apr. 2013.