

# Connecting UI and Business Processes in a Collaborative Sketching Environment

Markus Kleffmann

markus.kleffmann@paluno.uni-due.de

Marc Hesenius

marc.hesenius@paluno.uni-due.de

Volker Gruhn

volker.gruhn@paluno.uni-due.de

paluno - The Ruhr Institute for Software Technology  
University of Duisburg-Essen  
Gerlingstr. 16, 45127 Essen, Germany

## ABSTRACT

Sketching is an important activity in software development projects and has many advantages over strict formal languages, especially in cross-functional teams with different technical background. Sketches are used to develop all kinds of diagrams, providing a different view on the application and the underlying business logic. Several tools have been developed in recent years to support sketching activities, especially in the area of UI prototype sketching. Sketches are used to quickly develop a first impression of future UIs, helping designers, engineers, and domain experts to gain a common understanding of layout and dialog flows. Unfortunately, UI sketching tools focus on a single application aspect – the UI – and do not take business processes, technical data structures, and their relationships into account. We demonstrate how UI, business process, and technical diagram sketches can be interconnected in an augmented team room.

## ACM Classification Keywords

D.2.2 Design Tools and Techniques: Computer-aided software engineering (CASE); H.5.3 Group and Organization Interfaces: Computer-supported cooperative work

## Author Keywords

Cooperative Design; Electronic Whiteboards; Sketches

## INTRODUCTION

Studies have shown the importance of sketches in design and engineering disciplines [12, 20]. Since modeling languages are often too formal for early design phases [22], collaborative and informal sketching is highly beneficial for stakeholders [4] and classical CASE tools are mostly used in later project phases to document mature and final models. Stakeholders require an overview of the static and dynamic aspects of a project and sketch business processes and technical diagrams to visualize the relationships between business and technical data entities.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [Permissions@acm.org](mailto:Permissions@acm.org).

EICS'15, June 23–26, 2015, Duisburg, Germany.

Copyright is held by the owner/author(s). Publication rights licensed to ACM.

ACM 978-1-4503-3646-8/15/06...\$15.00

DOI: <http://dx.doi.org/10.1145/2774225.2775076>

Creating User Interface (UI) sketches is equally important when developing interactive applications [11]. Most designers prefer hand-drawn sketches when creating UI drafts [1] and both engineers and designers typically use whiteboards or flip charts [4, 5, 18]. Pure physical whiteboards and flip charts unfortunately have many disadvantages (e.g. limited canvas space, laborious digitizing, no versioning etc.), hence different informal sketching and prototyping tools for large interactive displays exist. To the best of our knowledge, no approach automatically integrates and visualizes the relationship between different sketch types. We argue that considering and using these relationships, especially between UI sketches and process/structural sketches, is highly beneficial to stakeholders for several reasons:

- Stakeholders need to navigate frequently between related artifacts while working on the same task [17, 18]. Making navigation as intuitive as possible is crucial for the workflow to keep up with thought processes [8].
- Artifacts are often semantically interconnected but stored as physically separated files that cannot be located ad hoc. Identification of relevant artifacts and navigation among them is therefore difficult, especially for stakeholders without extensive project knowledge.
- Changing tools interrupts workflows. Consequently, creating all relevant sketch types within the same tool is crucial, but most prototyping tools focus on storyboards and UI sketches and do not allow to sketch and combine business processes and data structures.
- Separated artifacts can quickly become inconsistent. It is important to support stakeholders in uncovering inconsistencies and contradictions in diagrams and sketches.

We discuss how the Augmented Interaction Room (AugIR), a physical team room equipped with wall-sized touchscreens, supports stakeholders in UI design activities by automatically connecting related UI, business process, and technical diagram sketches. Furthermore, we demonstrate how potential inconsistencies and contradictions in related sketches can be unveiled.

We start with a review of related work, followed by a description of the AugIR's key concepts and a discussion of our solution approach. Finally, we discuss our first validation, before concluding the paper with an outlook on future research.

## RELATED WORK

Several sketching and prototyping approaches have been developed in recent years, most of them favoring an informal approach and avoiding strict formal languages to not interrupt the creative workflow.

For example, Chen, Grundy and Hosking [4] introduce a modeling tool that uses electronic whiteboards for free-hand diagram sketching in early project phases. The sketches can be transformed into computer-drawn diagrams and exported to other CASE tools for further use.

Mangano and van der Hoek [14] present Calico, a sketching tool that is especially suited for modeling during the early phases of a software development project. It allows the designers to create free-hand sketches on electronic whiteboards or tablets.

West et al. [26] present an application called FlexiSketch which is a sketching tool for free-hand modeling, especially developed for the use on mobile devices.

Approaches that focus on UI sketching usually do not take other diagram types (e.g. business processes or object maps) into account.

Bailey and Konstan [1] for example present DEMAIS, a tablet-based application that strives to support designers in creating and subsequently executing behavioral UI sketches. They successfully prove that informal approaches have many advantages over strict formal languages and are favored by users, but also mention improvement possibilities, especially the lack of visualizing structural relationships between storyboards.

Lin et al. [13] present DENIM, a visual language for creating interactive web sites. They offer reusable sketch components and provide different granularity levels, e.g. a site map, storyboards and individual web pages. Furthermore, they elaborate how interactive elements changing a site's state (such as checkboxes) can be handled, but focus solely on UI aspects.

We observed that only a small number of research projects and commercial products allow the creation of usable connections between different sketches and diagrams. Noticeable, all approaches that we found are restricted to the creation of manual connections and thus depend on stakeholders to detect the relationships themselves.

Memmel and Reiterer [16] for example introduce a tool called Inspector and present a UI specification technique for interactive systems, that allows stakeholders to create relationships between graphical user interfaces, requirements and business processes on different abstraction levels. However, these relationships have to be created manually by the users.

Sparx Enterprise Architect [25] is a well-known commercial tool for modeling business and IT systems in various languages like UML, SysML, and BPMN. It can be used to create trace links between different kinds of artifacts and UI mockups, especially focusing on the impact analysis and visualization of relationships. The tool follows a very for-

mal approach and trace links have to be created manually by adding typified edges between model elements.

No sketching tool – to the best of our knowledge – automatically creates usable connections between related sketches.

## THE AUGMENTED INTERACTION ROOM

Engineers and designers prefer free-hand drawing and writing in early design and modeling phases [5, 7]. Consequently, the AugIR enforces informal and incomplete sketches and avoids strict modeling languages. Hand-drawn elements and written text are continuously analyzed using pattern and handwriting recognition to automatically create so-called *trace links* between elements that enable relationship visualization and navigation among related sketches.

Whereas other approaches use exactly one large interactive display, the AugIR uses several connected wall-sized displays and explicitly considers the relationships between them, thus offering a much more extensive view onto the software project. Each display in the AugIR visualizes a certain perspective and shows exactly one sketch (e.g. business process, structural diagram, UI etc.). Additionally, an arbitrary number of related artifacts, such as design documents or bug reports, can be superimposed on each wall and edited in place.

Various different perspectives are offered, such as a project overview that visualizes all sketches and their relationships as a graph, an integration map that illustrates the interfaces and dependencies with surrounding systems, and a migration map that helps stakeholders to manage the complexity of data transfers from legacy systems. In this paper however, we focus on the following three perspectives:

**Process Map** Dedicated to the business domain's dynamic aspects, the Process Map visualizes relevant parts of the business processes and provides an overview of the system's main functional requirements.

**Object Map** Displays the business domain's static aspects, i.e. the entities and artifacts the processes work on.

**Interaction Map** Used to sketch the UI, including the project's navigation structure and the look and feel of its key dialogs.

Figure 1 shows our current prototype used to model data structures, business processes, and UIs of an online shop. The right display is used to sketch the dialog where customers can enter their payment details while the left display shows the corresponding business process sketch. A possible four-wall layout could include three walls for modeling the structural, behavioral, and interactive aspects of the system and a fourth wall for the project overview.

Stakeholders can mark elements on sketches with special symbols, so-called *Annotations*. Each annotation has its unique visualization and semantic. During moderated workshops, stakeholders place annotations in sketches to highlight process or system elements that require special attention, e.g. because they implement key business aspects, are very complex, or not well understood. Discussions usually focus on annotated elements and related aspects, helping stakeholders

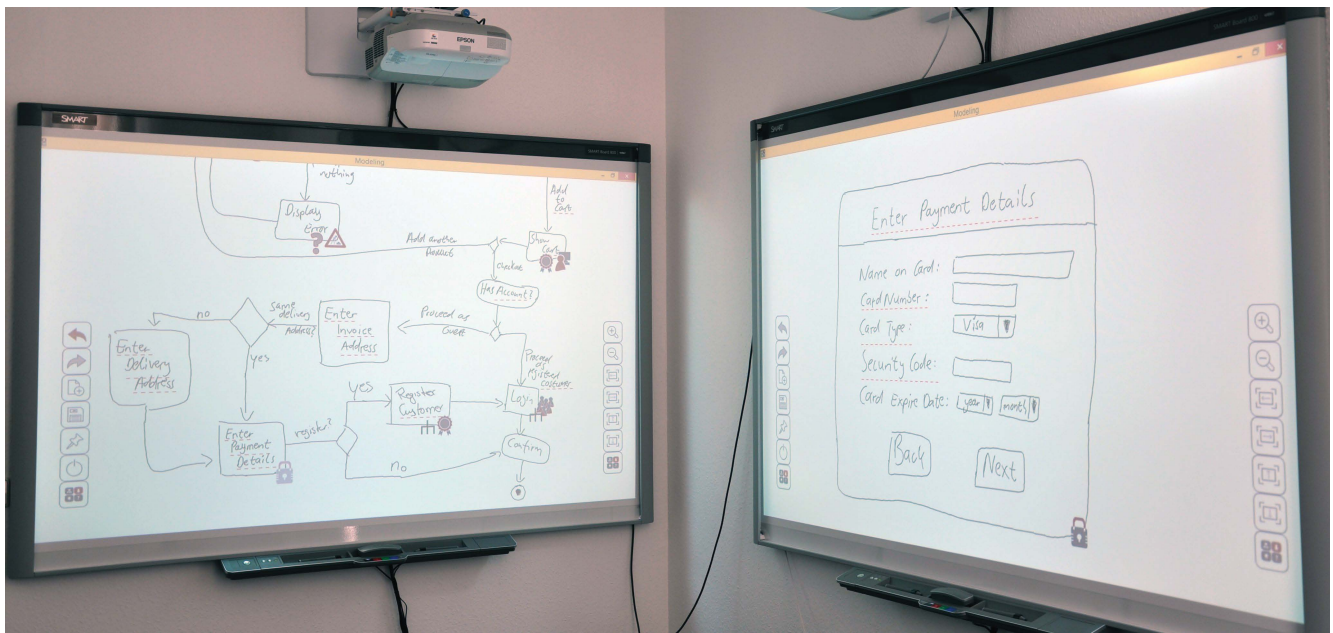


Figure 1. Prototype of the Augmented Interaction Room (AugIR).

to develop a common understanding of the project's most important risk and value drivers. In contrast to simple textual notes, annotations allow for direct semantical characterization and highlight important aspects.

Table 1 shows four frequently used annotations (out of a total set of 23): *Revision Necessary* denotes elements for future revision and possible change; *Immutable* elements must not be changed or modified in any way, e.g. interfaces adopted from legacy systems; *Usability* denotes particularly complex or important dialogs; and *Security* highlights possible security requirements, e.g. components processing sensitive user data.

### CONNECTING UI AND RELATED PROCESSES

In this section, we discuss how trace links are created in the AugIR, how these links are visualized and used for navigation between related artifacts, and how they support stakeholders in avoiding inconsistencies and potential contradictions.

#### Trace Link Creation

In projects conducted within the AugIR, the Interaction Map is used to sketch the UI of an application, its navigation structure, and the look and feel of its key dialogs. The Interaction Map provides two levels of refinement, switchable at any time: the storyboard level and the sketch level. On the storyboard level, the Interaction Map presents an overview of





all created dialog sketches and their relationships as a graph. Stakeholders can create several storyboards, e.g. one storyboard for each key business process, and interconnect them to visualize the complete application dialog flow. Figure 1 shows the sketch level, where the Interaction Map displays one single dialog sketch to be modified or annotated.

Trace links provide connections and relationships between UI, business process and structural sketches, visualizing how the same business entity is represented in the various perspectives provided by each map. They represent an entry point to another view on the project from the entity of interest. The AugIR uses a combination of information retrieval and fuzzy search techniques [10] to recover trace links automatically and unnoticeable for stakeholders:

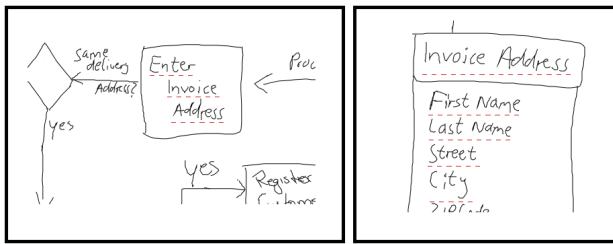
We denote each hand-drawn element (e.g. an activity in a business process sketch or an object in an object diagram sketch) as a triple  $e = \{t, s, A\}$ , where  $t$  is the text of the element (consisting of one or more words separated by blanks),  $s$  is its shape, and  $A$  is the set of annotations applied. For example, a process activity "Sign contract" is a sketch element in a business process sketch consisting of the words "Sign" and "contract", a rectangular shape around the text, and a set of applied annotations. Notice that sketches may contain text without a shape (e.g. textual notes in a sketch) as well as shapes without a text (e.g. certain symbols like diamonds and arrows). The latter are ignored by our algorithm, since we are only interested in elements containing textual information. The set  $A$  is empty if no annotations have been applied.

Trace links between sketches are created by analyzing the sketch elements and their textual content. When stakeholders draw a new element  $e = \{t, s, A\}$  with  $t = w_1, \dots, w_n$ , we create a search vector  $q_i := (q_{i1}, \dots, q_{in})$  for each sketch

Table 1. Examples of Interaction Room annotations.

Symbol				
Meaning	Revision Necessary	Immutable	Usability	Security





**Figure 2.** Example for visualization of trace links via dashed underlines: A trace link between a process sketch (left) and an object diagram sketch (right), based on the words "Invoice" and "Address".

$S_i$  in the project repository as follows: If  $w_j$  occurs in any sketch element of  $S_i$ , we set  $q_{ij} := w_j$ . Otherwise, we test if a word  $w'_j$  in  $S_i$  exists that is sufficiently similar to  $w_j$ . If so, we set  $q_{ij} := w'_j$ ; if not, we set  $q_{ij} := w_j$ . Replacing a word  $w_j$  with a sufficiently similar word  $w'_j$  can be necessary to achieve best search results, because we are analyzing handwritten text and the results may contain falsely recognized characters. We use a modified Levenshtein distance to compute the similarity of two words and afterwards apply Latent Semantic Indexing (LSI) using the formerly created search vectors  $q_i$  to create trace links between  $e$  and matching sketch elements from other sketches.

### Trace Link Visualization and Navigation

Finding an appropriate visualization for trace links is a non-trivial task [15]. We strive to visualize the (often implicit and therefore easily overlooked) relationships directly within the sketches, making them usable for stakeholders and enabling easy and intuitive navigation to support the usually frequent focus shifts during design activities. Instead of laboriously navigating across file names and directories, stakeholders are able to navigate among related sketches and documents via intuitive gestures using the semantics of their content and the inherent relationships between them. Consequently, stakeholders are allowed to focus on the task at hand – modeling the developed application – instead of the used tool chain.

A small dashed line located directly under a sketch element indicates a trace link (see Figure 2). Tapping on such an underlined element either opens a floating window displaying the related sketch (if there is only one) or shows a small preview set of all related sketches (if there are several). When opening other sketches via trace links, related elements are specifically highlighted to visualize the relationship, so stakeholders gain a direct hint on how the element is used in this perspective. These trace links provide valuable information about business entities, their relationships and their roles in different perspectives. Without a proper trace link visualization, these relationships would easily be missed by stakeholders without deeper project knowledge.

The same approach is used to visualize trace links between sketches and related text documents (e.g. specification documents). Tapping on an underlined element will show the document's content in a floating window and also allows stakeholders to edit it in place. If there are several related documents, tapping on the underlined sketch element will show a

list with small previews of the linked documents and stakeholders can select the desired one.

Trace link visualization is not only used as an indicator for the existence of a dependency or relationship (which would possibly remain hidden otherwise), but can also be used to easily access the content of a corresponding artifact without requiring the stakeholders to know its exact location in the project repository. By using dashed underlines, trace links are always visualized near the elements to which they actually apply. Furthermore, their appearance reminds the stakeholders of the well-known concept of hyperlinks. Therefore, this approach is intuitively comprehensible for stakeholders and integrates seamlessly into their workflow.

### Indirect Annotations

Based on the recovered trace links, the AugIR's control software continuously performs a so-called impact analysis, i.e. it analyzes changes, detects possible inconsistencies, and reacts accordingly. While impact analysis is generally considered a non-trivial task for large software projects [19, 24], two important AugIR features add to the complexity: various simultaneous views on the project requiring consistency and annotations conveying additional meta-information that may become inconsistent or lead to contradictions. The AugIR's extensive impact and relationship analysis counters these issues and strives to assist stakeholders in several ways.

To identify possible inconsistencies and contradictions, the AugIR monitors all annotation activities. Whenever an annotation is applied to a sketch element, the AugIR identifies related sketches, i.e. sketches also containing an element of this name. It tests whether the used annotation is also applicable to the regarded element within the related sketch(es) or if it contradicts any of the annotations applied there. If the annotation is applicable, it is also applied to all occurrences of the regarded element in the related sketch(es) to avoid inconsistencies. We call annotations transferred from and to other sketches *Indirect Annotations* and visualize them slightly transparent.

In addition to countering potential inconsistencies, indirect annotations also provide a beneficial overview of all annotations set for a specific element in any sketch. Since multiple stakeholders create and annotate sketches during application development, indirect annotations are a valuable asset – they might uncover new aspects resulting in and from stakeholder discussions over certain elements in different perspectives.

Figure 3 demonstrates why indirect annotations are useful: when creating the UI for a dialog where customers enter their credit card details, stakeholders were aware that transmitting and storing payment data requires high security, hence they annotated the dialog with the annotation *Security*. Later on, when the business process was sketched, the AugIR automatically annotated the activity "Enter Payment Details" with an indirect *Security* annotation due to the trace link between this process step and the UI. Thus, the relationship between UI and business process was directly visible to participating stakeholders.

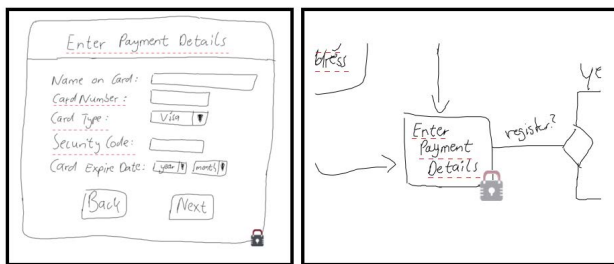


Figure 3. Example: The "Security" annotation has been placed in a GUI sketch (left) and is visualized slightly transparent as an indirect annotation on the corresponding activity in a process sketch (right).

If an annotation leads to a contradiction, the AugIR presents a warning and advises stakeholders to either not use this specific annotation or to modify the conflicting locations accordingly. For example, adding the annotation *Revision Necessary* to an element that is already annotated as *Immutable* implicates a contradiction. While contradictions can be identified easily when applying opposing annotations to the same element within the same sketch, uncovering them when working with different sketches is more difficult without technical support. However, stakeholders may ignore warnings and apply contradicting annotations anyway, because the AugIR does not restrict the stakeholders' design and modeling activities. Instead, it strives to support stakeholders in their collaborative work by uncovering potential issues that are overlooked otherwise.

## EVALUATION & DISCUSSION

The techniques described in this paper are implemented in the AugIR prototype shown in Figure 1. The control software is developed in C# using the Windows Presentation Foundation (WPF). We use several interactive SMART Boards [23] as wall-sized displays. Our pattern recognition approach is based on work by Coyette et al. [6], while handwriting recognition is implemented by using the Microsoft.Ink libraries built into the .NET framework.

Although the AugIR prototype is ongoing work and still under development, a first version was deployed at a German consulting company for internal projects and evaluation over a period of three months. During this time, the AugIR was used in ten sessions and stakeholders mostly worked on process maps, object maps and UI sketches. Each session lasted approximately three hours and five stakeholders participated on average. The initial feedback was very positive – stakeholders especially rated the easy navigation between artifacts and the intuitive refinement of sketches as beneficial and supportive.

However, two problems still exist in the algorithm previously presented: Firstly, false positive trace links may be created if sketch elements contain the same words in different order. For example, the process activities "Offer review" and "Review offer" share the same words, although they have a different meaning. Consequently, our algorithm would create a false positive trace link between the corresponding sketch elements. To solve this issue, we must analyze and consider

the order in which words appear. Secondly, our algorithm can not yet determine synonyms. Therefore, it is unable to create a trace link between elements that have the same meaning but use different words, e.g. "Prepare offer" and "Prepare proposal". Divergent denotations may occur in praxis because different sketches are often created by different stakeholders.

An extensive formal validation that especially focuses on analyzing the quality of the automatically created trace links will shortly follow. We will begin with multiple quantitative evaluations in isolated experiments to optimize the recovery results and minimize the amount of falsely created trace links. The validation will be concluded with qualitative evaluations in different industry projects, especially considering the issues discussed above. The AugIR's non-augmented sibling based on physical whiteboards and magnetic annotations, the Interaction Room (IR), has already been evaluated in several industry projects [2, 3, 9]. As soon as the AugIR reaches a sufficient maturity level, we will gradually introduce the technology in these projects. Closely observing the projects in which this substitution takes place and performing interviews with the stakeholders on a regular basis will allow us to evaluate the efficiency and practical benefits of our approach in general and the concepts presented above in particular.

## CONCLUSION & FUTURE WORK

In this paper, we presented how different project perspectives sketched in the AugIR – a project room based on interactive wall-sized displays – can be connected via automatically generated trace links to provide an overall understanding of the relationships in a software development project. We demonstrated how the same business entities can be visualized and interconnected in UI sketches, business processes and technical data structures. We contributed to the field of engineering interactive computing systems by demonstrating how the AugIR can support stakeholders in revealing relationships, inconsistencies and contradictions in different project sketches.

For future work, more improvements to the interaction map are planned to simplify interactive application development. With the advances in Human Computer Interaction (HCI), more sophisticated interaction paradigms than classic mouse and keyboard become interesting for mainstream application developers. Consequently, tools aiming to support stakeholders must integrate e.g. touch gestures or voice control into UI sketching applications.

Different stakeholders might work on different application aspects in several session. Consequently, different wording and naming for entities and actions might be found in different sketches. We plan on enabling the AugIR to automatically recognize synonyms and create corresponding trace links.

The use of wall-sized touchscreens is only a first step. The integration of further devices such as cameras, voice control or special augmented-reality glasses is conceivable and seems beneficial. Similar approaches already exist that promote for example the use of eye tracking to improve trace link recovery and maintenance [21]. Such techniques could easily be adopted for the AugIR and promise to further enhance its capabilities and usefulness.

## REFERENCES

1. Bailey, B. P., and Konstan, J. A. Are informal tools better? comparing demais, pencil and paper, and authorware for early multimedia design. In *Proc. SIGCHI Conf. Human Factors in Computing Systems*, CHI '03, ACM (New York, NY, USA, 2003), 313–320.
2. Book, M., Grapenthin, S., and Gruhn, V. Risk-aware migration of legacy data structures. In *Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference on* (2013), 53–56.
3. Book, M., Grapenthin, S., and Gruhn, V. Value-based migration of legacy data structures. In *Software Quality. Model-Based Approaches for Advanced Software and Systems Engineering*, vol. 166. Springer International Publishing, 2014, 115–134.
4. Chen, Q., Grundy, J., and Hosking, J. An e-whiteboard application to support early design-stage sketching of uml diagrams. In *Human Centric Computing Languages and Environments, 2003. Proceedings. 2003 IEEE Symposium on* (2003), 219–226.
5. Cherubini, M., Venolia, G., DeLine, R., and Ko, A. J. Let's go to the whiteboard: how and why software developers use drawings. In *Proc. SIGCHI Conf. Human Factors in Computing Systems* (2007), 557–566.
6. Coyette, A., Schimke, S., Vanderdonckt, J., and Vielhauer, C. Trainable sketch recognizer for graphical user interface design. In *Proc. 11th IFIP TC 13 Intl. Conf. Human-computer Interaction* (2007), 124–135.
7. Dekel, U., and Herbsleb, J. D. Notation and representation in collaborative object-oriented design: an observational study. *SIGPLAN Not.* 42, 10 (Oct. 2007), 261–280.
8. Ferguson, E. S. *Engineering and the Mind's Eye*. MIT Press, 1994.
9. Grapenthin, S., Book, M., Gruhn, V., Schneider, C., and Völker, K. Reducing complexity using an interaction room: An experience report. In *Proc. 31st ACM Intl. Conf. Design of Communication, SIGDOC '13*, ACM (New York, NY, USA, 2013), 71–76.
10. Kleffmann, M., Röhl, S., Book, M., and Gruhn, V. Establishing and navigating trace links between elements of informal diagram sketches. In *8th International Symposium on Software and Systems Traceability* (2015). to appear.
11. Landay, J. A., and Myers, B. A. Sketching interfaces: Toward more human interface design. *Computer* 34, 3 (Mar. 2001), 56–64.
12. Larkin, J. H., and Simon, H. A. Why a diagram is (sometimes) worth ten thousand words. *Cognitive Science* 11, 1 (1987), 65 – 100.
13. Lin, J., Thomsen, M., and Landay, J. A. A visual language for sketching large and complex interactive designs. In *Proc. SIGCHI Conf. Human Factors in Computing Systems* (2002), 307–314.
14. Mangano, N., and van der Hoek, A. The design and evaluation of a tool to support software designers at the whiteboard. *Automated Software Engineering* 19, 4 (2012), 381–421.
15. Marcus, A., Xie, X., and Poshyvanyk, D. When and how to visualize traceability links? In *Proceedings of the 3rd international workshop on Traceability in emerging forms of software engineering* (2005), 56–61.
16. Memmel, T., and Reiterer, H. Inspector - interactive ui specification tool. In *In Proc. of the 7th International Conference On Computer Aided Design of User Interfaces (CADUI) 2008*, Springer (2008), 161–174.
17. Myers, B., Park, S. Y., Nakano, Y., Mueller, G., and Ko, A. How designers design and program interactive behaviors. In *Visual Languages and Human-Centric Computing, 2008. VL/HCC 2008. IEEE Symposium on* (Sept 2008), 177–184.
18. Petre, M. Insights from expert software design practice. In *Proceedings of the the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering, ESEC/FSE '09*, ACM (New York, NY, USA, 2009), 233–242.
19. Sapna, P., and Mohanty, H. Ensuring consistency in relational repository of uml models. In *Information Technology, (ICIT 2007). 10th Intl. Conf. on* (2007), 217 –222.
20. Schütze, M., Sachse, P., and Römer, A. Support value of sketching in the design process. *Research in Engineering Design* 14 (2003), 88–97.
21. Sharif, B., and Kagdi, H. On the use of eye tracking in software traceability. In *Proceedings of the 6th International Workshop on Traceability in Emerging Forms of Software Engineering, TEFSE '11*, ACM (New York, NY, USA, 2011), 67–70.
22. Shipman, F. M., and Marshall, C. C. Formality considered harmful: Experiences, emerging themes, and directions on the use of formal representations in interactive systems. *Computer Supported Cooperative Work (CSCW)* 8, 4 (1999), 333–352.
23. Smart Technologies. Smart board. <http://smarttech.com/smartboard>.
24. Spanoudakis, G., and Zisman, A. Inconsistency management in software engineering: Survey and open research issues. In *Handbook of Software Engineering and Knowledge Engineering*, World Scientific (2001), 329–380.
25. Sparx Systems. Sparx enterprise architect. <http://www.sparxsystems.com>.
26. Wüest, D., Seyff, N., and Glinz, M. Flexisketch: A mobile sketching tool for software modeling. In *Mobile Computing, Applications, and Services*, vol. 110. Springer Berlin Heidelberg, 2013, 225–244.