

Establishing and Navigating Trace Links Between Elements of Informal Diagram Sketches

Markus Kleffmann, Sebastian Röhl, Volker Gruhn
 paluno – The Ruhr Institute for Software Technology
 University of Duisburg-Essen, Germany
 {markus.kleffmann, sebastian.roehl, volker.gruhn}
 @paluno.uni-due.de

Matthias Book
 Dept. of Computer Science
 University of Iceland
 book@hi.is

Abstract—The Augmented Interaction Room is a team room whose walls are outfitted with wall-sized touchscreens that are used to create informal diagram sketches. It strives to help stakeholders in fostering a common understanding of a project’s most important risk and value drivers by visualizing aspects of a software system from different perspectives and allowing stakeholders to annotate elements that are especially important or critical for project success. In this paper, we discuss how a combination of traceability techniques and fuzzy search methods can be used to efficiently support the stakeholders’ collaboration and their early design and decision making activities by providing an easy and intuitive visualization and navigation approach and by automatically uncovering potential inconsistencies and contradictions in the created sketches. Since the pragmatic methodology of the Augmented Interaction Room encourages stakeholders to work with handwritten sketches rather than with formal modeling languages, the identification and maintenance of trace links is particularly challenging.

I. INTRODUCTION

Studies have shown the importance of diagram sketches in software engineering and other design and engineering disciplines [15], [23], [34], [43]. Collaborative sketching is especially important in the early phases of a software development process [5]. Since modeling languages are often too formal for early design phases [36], classical CASE tools are mostly used for documentation of mature or final models, and most engineers and designers prefer to perform their modeling activities on whiteboards or flip charts [5], [8], [10], [25], [31].

Since pure physical whiteboards and flip charts introduce a variety of issues [5], [11], [12] (e.g. limited canvas space, no drawing support, laborious digitizing, no versioning etc.), we presented the idea of an Augmented Interaction Room (AugIR) in our previous works [18], [21], [20]. The AugIR is a physical team room that helps stakeholders to foster a common understanding of the project’s most important risk and value drivers. Its walls are outfitted with large wall-mounted touchscreens (see Fig. 1) that are used for collaboratively creating diagram sketches, such as process models, class diagrams or migration maps, which are directly drawn onto the surfaces of these touchscreens using special pens. We intentionally use free-hand drawing and writing instead of the often heavyweight techniques of classical CASE tools, because studies have shown that it is much more efficient for diagram sketching, and that informal notations are preferred by

most users over formal graphical modeling languages [8], [12]. Such a pragmatic approach also reduces technology barriers, especially for non-technical stakeholders whose inclusion into the modeling process is one of the key aims of the Interaction Room approach.

Stakeholders can annotate the diagram sketches with markers, so-called *Annotations*, which are graphical symbols that have specific semantics (see table I). Annotations are used to explicitly indicate and discuss elements that are especially important to certain stakeholders, highlight aspects that are critical for project success, uncover uncertainties of individual team members and make implicit project knowledge explicit.

In this paper, we discuss how a combination of traceability techniques and fuzzy search methods can be used to support the stakeholders’ collaboration and their early design and decision making activities by providing an easy and intuitive visualization and navigation approach and by automatically uncovering potential inconsistencies and contradictions in the created sketches.

First of all, engineers and designers need to switch their focus frequently while working on the same task [30], [31], [46]. This requires frequent changes of abstraction levels [8] as well as navigation between related diagrams [30], [31]. We call the first type of navigation *vertical navigation* and the second type *horizontal navigation*. It is crucial to support both types of navigation and to make them as intuitive as possible, to allow the stakeholders to work fast enough to keep up with their thought process [13], [32]. How we can support stakeholders in navigating vertically has already been discussed in our previous work [19]. Therefore, this paper focuses on the horizontal navigation between related diagram sketches, based on recovered trace links between them.

Furthermore, complex software projects usually consist of many logically and physically separated artifacts between which designers and engineers have to navigate frequently. However, these artifacts can often not be located ad hoc in a large project repository (if at all), or require a laborious search. A reason for this problem is that often a multitude of implicit relationships between the artifacts exists, which are neither obvious nor explicit. This makes identification of relevant artifacts and navigation among them often difficult, especially for stakeholders without extensive project knowledge. There-

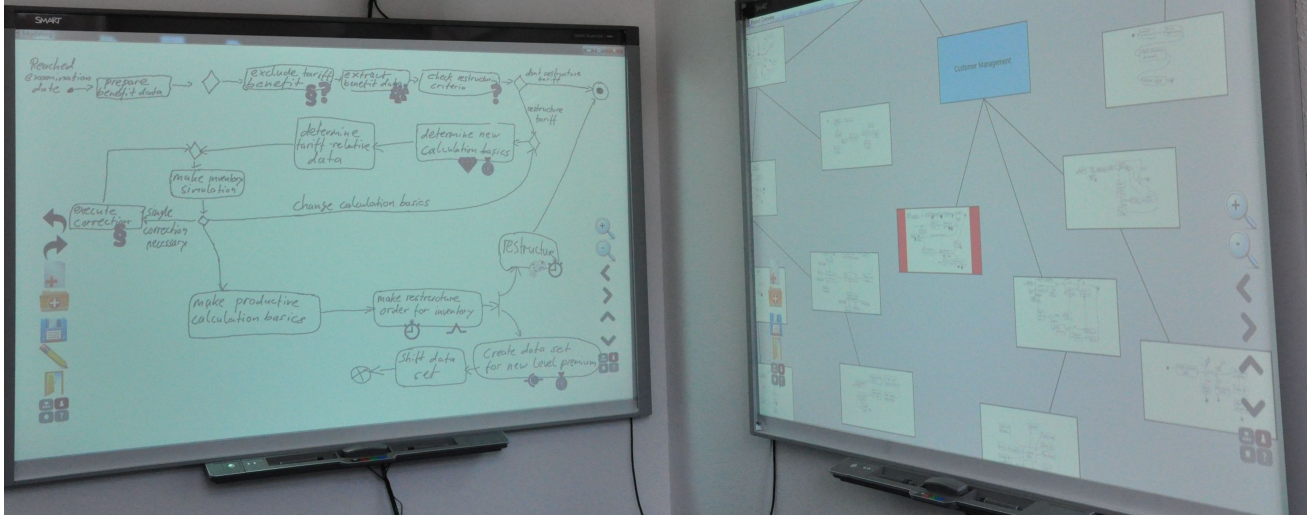


Figure 1. Prototype of the Augmented Interaction Room (AugIR). The left display shows a hand-drawn process sketch, the right display shows previews of all sketches in the project and their refinements.

fore, it seems beneficial to explicitly visualize those hidden relationships and make them usable, e.g. for navigation to related artifacts.

Another common software engineering problem is that different diagram sketches can quickly become inconsistent. In the AugIR, this issue is especially critical because it does not only apply to the sketched diagrams but also to the added annotations which provide a lot of meta-information for the sketches. Therefore, it is also important to support the stakeholders in uncovering potential inconsistencies and contradictions in the created sketches.

This paper begins with a brief overview of important related work in section II, followed by a description of the key concepts of the AugIR in section III. Section IV discusses our solution approach in detail. In section V, we present the prototype that we are currently working on and briefly discuss our first internal validation, before we conclude with an outlook on future work in section VI.

II. RELATED WORK

Several companies and research centers are experimenting with collocating teams in rooms, so-called “war rooms” or project rooms [29]. In almost any scenario, whiteboards and flip charts are frequently used for collaborative design activities [8], or visualization of important project knowledge [10].

Replacing physical whiteboards with large interactive displays can be very beneficial for these team rooms, as for example shown by Streitz et al. [41] and Haller et al. [16]. While these approaches focus on collaborative work in general, there also exist various approaches that specifically regard the design and modeling activities on large interactive displays.

For example, Chen, Grundy and Hosking [6] introduce a UML 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 et al. [25] 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.

Glinz et al. [45] present an application called FlexiSketch which is a sketching tool for free-hand modeling, especially developed for the use on mobile devices.

We did not find any approaches that explicitly regard the usage of traceability techniques and the benefits that can be achieved by automatically creating relationships between the sketches.

In general, one can distinguish between prospective and retrospective traceability [2], [9]. While prospective traceability techniques observe the activities of users and capture trace links between artifacts in situ while users work with them, retrospective traceability techniques generally analyze a static set of existing artifacts and recover trace links between them based on the extracted information. One can effectively combine different techniques to employ the strengths of several approaches, as already demonstrated by Asuncion, Asuncion and Taylor [2], Chen, Hosking and Grundy [7] and others.

Many approaches use Latent Semantic Indexing (LSI), which is an extension of the Vector Space Model (VSM). LSI is an information retrieval method that analyzes the similarity of textual documents by identifying patterns in their relationships. For example, Marcus, Maletic and Sergeyev [27] use LSI to create trace links between source code and free text documents, such as requirements or design documents. Antoniol et al. [1] apply the VSM for the same purpose.

Another interesting approach is presented by Spanoudakis [39], where trace links between textual requirement documents and UML object models are created automatically by using

heuristic traceability rules that describe how textual parts in the requirement documents match related elements in the object models.

In Sect. IV, we will discuss how our approach combines LSI with a fuzzy search metric to create usable relationships between artifacts in the AugIR.

III. THE AUGMENTED INTERACTION ROOM

As motivated in Sect. I, engineers and designers generally prefer free-hand drawing and writing in the early design and modeling phases. As a result, all sketches in the AugIR can remain informal and incomplete, as we do not enforce the use of a strict formal modeling language. Nevertheless, the AugIR continuously analyzes all hand-drawn elements and written text using pattern and handwriting recognition. The artifact relationships described in this paper are mostly based on these results.





While other approaches described in Sect. II also use large interactive displays, they mostly only regard one single large display. In contrast to that, our approach explicitly considers the relationships between several connected wall-sized displays. This offers a much more extensive view onto the software project by providing different perspectives for a particular component or process at the same time—a feature that especially benefits engineers who often require multiple different views when designing a complex software architecture [15].

Each display in the AugIR visualizes a certain perspective onto the system and shows exactly one diagram sketch at any time (e.g. process sketch, structural diagram sketch etc.). Additionally, an arbitrary number of related artifacts, such as design documents, screenshots, or bug reports, can be superimposed on each wall and edited in place.

Figure 1 shows an example from our current prototype that models the data structures and business processes of an insurance company. In this example, the display on the left wall is used to draw a process sketch while the right wall displays an overview of the project, the contained sketches and their refinements for vertical navigation. A possible four-wall AugIR layout could include three walls for modeling the structural, behavioral and interactive aspects of the system, as well as a fourth wall for the project overview.

Stakeholders can annotate diagrams and sketches with a variety of annotations. Each annotation has its unique symbol and a specific semantic. During moderated workshops, these annotations can be placed in the diagram sketches to highlight process or system elements that require special attention, e.g. because they implement key business aspects that are particularly complex or not yet well understood. Discussions in the Interaction Room usually revolve around these annotated elements in particular, which helps stakeholders to foster a common understanding of the project’s most important risk and value drivers. In contrast to simple textual notes, as they are available in most sketch tools, annotations allow for a much more semantically characterization and easier understanding of important aspects.

Table I
EXAMPLES OF INTERACTION ROOM ANNOTATIONS.

| Symbol |  |  |  |  |
|---------|--|---|---|---|
| Meaning | Revision Necessary | Immutable | Uncertainty | Security |

The Interaction Room method defines a total number of 23 different annotations that can be used within the diagram sketches. Table I shows four examples of frequently used annotations: The “*Revision Necessary*” annotation denotes elements that need to be revised or changed in the near future. The “*Immutable*” annotation denotes elements that must not be changed or modified in any way, e.g. certain interfaces that are adopted from a legacy system. The “*Uncertainty*” annotation expresses uncertainty, e.g. if a process is not yet well understood or insufficiently specified. The “*Security*” annotation denotes elements that bear particular security requirements, e.g. components that access sensitive user data or passwords.

Contradictions may arise by applying annotations with opposite semantics. For example, if an element is already annotated as “*Immutable*”, it must usually not be annotated with the “*Revision Necessary*” annotation at the same time. While such contradictions can easily be identified when applying opposing annotations to the same element within the same sketch, uncovering them when working with different artifacts that contain the same hand drawn elements is much more difficult.

IV. SOLUTION APPROACH

To address the issues described in the previous sections, we strive to automatically create explicitly visualized links between related artifacts that make implicit (and therefore often overlooked) relationships obvious and usable.

We use a graph to store and describe the relationships between sketches as well as between sketches and related documents (e.g. bug reports, screenshots, specification documents etc.). An example of such a graph that displays the refinements for vertical navigation is shown on the right display in Fig. 1. The concept is based on so-called mega- or macromodels that contain models and their relationships [17]. We described the meta model for this graph in detail in our previous work [19].

In the following, we discuss how trace links are recovered in the AugIR, how they enable stakeholders to navigate among related artifacts, and how they can help to counter certain inconsistencies and contradictions in the created sketches.

A. Creation of Trace Links

Trace links are created mostly automatically by using a combination of prospective and retrospective traceability techniques [18].

On the one hand, we automatically observe the stakeholders’ activities and capture trace links between artifacts in situ while the stakeholders work with them. For example, if the same documents are repeatedly opened in the context of the

same sketches, the AugIR creates trace links between them, assuming that they have a semantic relationship to each other. The more often these artifacts are displayed and worked on together in the AugIR, the stronger these trace links will become.

On the other hand, we analyze the existing sketches and documents and recover trace links between them based on the extracted textual information. For example, if there exists a process element named “Process offer” in a process sketch S_1 , and there exists an object named “Offer” in an object diagram sketch S_2 , the AugIR should be able to recover a trace link between these sketches S_1 and S_2 based on the related sketch elements.

To achieve this, the AugIR continuously analyzes all hand-drawn elements and written text in the background. The type of an element is determined by the kind of perspective in which the sketch is edited. For example, a rectangle drawn in a process sketch is interpreted as an activity, while a rectangle in an object diagram sketch is interpreted as an object. Whenever the handwriting recognition is able to recognize text, we try to identify the best matching artifacts for the drawn element.

Since we are analyzing handwritten text, the results may contain falsely recognized characters. Therefore, we have to apply fuzzy search techniques to counter such false recognitions before we can use traceability algorithms.

Our approach uses the Levenshtein distance [24], also known as edit distance, which is a well-known and often used metric for measuring the difference between two strings. It computes the minimum number of operations (i.e. character insertions, deletions or substitutions) that are required to change one string into the other [44]. We use this metric to compare the similarity of handwritten words with recognized words from other artifacts. If there is a match and the recognized words appear in a regarded artifact, we apply Latent Semantic Indexing (LSI), using the recognized words as search query and the terms from that artifact as term-document-matrix. If there is no exact match for one or more words, we compare the similarity of these words with all terms from the regarded artifact and eventually substitute them in the search query with the most similar ones (if the similarity is strong enough).

Furthermore, in our experiments we noticed that it is often difficult for the handwriting recognizer to distinguish certain similar looking characters, such as \mathfrak{t} and \mathfrak{f} , or \mathfrak{u} and \mathfrak{v} , because of their similar visual appearance. Therefore, we slightly modified the Levenshtein distance to counter such errors. Instead of computing an integer number, where each character substitution adds 1 to the total number of required operations, our modified method computes a floating point number, where character substitutions between very similar looking characters only add a fraction < 1 to the total number of required operations.

Using this modified Levenshtein distance, we compute the similarity of two words by using a formula that is based on the approach by Soukoreff and MacKenzie [38]. The similarity of

two words w_1 and w_2 is computed as

$$\sigma(w_1, w_2) := 1 - \frac{\text{LD}(w_1, w_2)}{\max(|w_1|, |w_2|)}$$

where LD computes the modified Levenshtein distance and $|w_1|$ and $|w_2|$ denote the lengths of the words w_1 and w_2 , respectively.

The resulting values range from 0 to 1. The more similar the words w_1 and w_2 are, the closer the value is to 1. A value of 1 indicates a perfect match while a value of 0 indicates that the words are completely different.

Based on this similarity measure, our algorithm works as follows:

Let S_1, \dots, S_n be the sketches in our project repository. Each sketch S_i contains one or multiple sketch elements, e.g. activities in a process sketch, objects in an object diagram sketch etc. The set of sketch elements contained in a sketch S_i is denoted as $E(S_i)$.

Each sketch element $e \in E(S_i)$ is defined as a tuple $e := (t, s)$, consisting of a text t and a shape s . A text t can consist of one or multiple words separated by a blank. For example, a process element “Sign contract” is a sketch element in a process sketch, consisting of the words “Sign” and “contract” and a rectangular shape around the text. 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 the following algorithm, since we are only interested in elements that contain textual information.

For each sketch S_i , we define a set $W(S_i)$ that contains all words that appear in any sketch element $e \in E(S_i)$.

Additionally, we define a matrix $M(S_i)$ that contains one row for each word $w \in W(S_i)$ and one column for each sketch element $e \in E(S_i)$. The matrix element m_{ij} at position (i, j) indicates how often the i th word appears in the j th sketch element.

When a new sketch element is drawn within a sketch S , the AugIR applies pattern and handwriting recognition to create an element $e = (t, s)$ with $t := w_1 w_2 \dots w_m$. The words w_1, w_2, \dots, w_m are used to create a vector $q := (w_1, w_2, \dots, w_m)$ that is used as a base for the search query.

For each sketch S_i with $S_i \neq S$ we create a vector $q_i := (q_{i1}, \dots, q_{im})$ by performing the following steps:

- 1) Test for each w_j with $j = 1, \dots, m$ if that word occurs in the list of words for sketch S_i , i.e. if $w_j \in W(S_i)$.
 - a) If $w_j \in W(S_i)$, set $q_{ij} := w_j$ and proceed with the next word of q (if there are any).
 - b) If $w_j \notin W(S_i)$, compute $\sigma_{\max} := \max\{\sigma(w_j, w) | w \in W(S_i)\}$. If σ_{\max} is greater than a certain threshold μ , set $q_{ij} := w$ for that $w \in W(S_i)$ with $\sigma(w_j, w) = \sigma_{\max}$. Otherwise set $q_{ij} := w_j$.
- 2) Apply Latent Semantic Indexing, using the vector q_i as search query and $M(S_i)$ as term-document-matrix. If the resulting value is above a certain threshold ν for

an element $e_i \in E(S_i)$, create a trace link between the elements e and e_i .

The technique described above can also be used to create trace links between sketches and other text documents in the project repository, e.g. bug reports or specification documents.

However, the optimal values for μ and ν have yet to be determined in future studies, in order to achieve the best results and to reduce the number of false positive trace links.

Furthermore, explicit trace links between sketches and artifacts can also be created manually by the users, since this approach cannot create retrospective trace links for non-textual project artifacts, such as screenshots [22].

B. Visualization and Navigation

Finding an appropriate visualization of recovered trace links is important and in most cases not trivial [28]. Our approach strives to visualize the (often implicit and therefore easily overlooked) relationships directly within the sketches, making them usable for the stakeholders, and enabling easy and intuitive navigation to support the frequent shifts of focus usually performed during design activities. Instead of laboriously navigating across file names and directories, the users should be able to navigate among related sketches and documents via intuitive gestures, using the semantics of their content and the relationships between them, allowing them to fully focus on the modeling process.

Obviously, stakeholders can open any sketch by tapping on its preview in the navigation graph (see Fig. 1). But more importantly, they can also navigate between related artifacts by using the trace links that are indicated with various icons (see Fig. 2):

The existence of a trace link between a sketch element and a document is visualized by a small stylized document symbol that is displayed next to the corresponding sketch element. Tapping on this icon opens a floating window that displays the document's content and also allows the designers to edit it in place (if supported by the document type). If there are several related documents, tapping on the icon will show a list with small previews of the linked documents instead, from which the stakeholders may select the desired one. Therefore, this icon 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 the corresponding document without requiring the stakeholders to know its exact location in the project repository. By always displaying the icons locally next to the elements to which they actually apply, this approach is intuitively comprehensible and should integrate seamlessly into the stakeholders' workflow.

Trace links between sketches are visualized in a similar manner, using a small stylized chain link symbol that is displayed next to the corresponding sketch element. Tapping on that icon opens a list with previews of all related sketches (if there are several), or directly shows the content of the related sketch (if there is only one). Additionally, the elements due to which the trace link exists are highlighted with a glowing

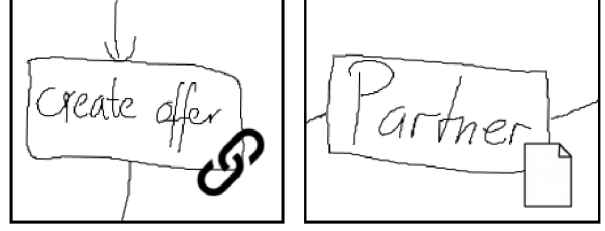


Figure 2. Examples for visualization of trace links. Left: Trace link to another sketch, based on the words “Create” and “offer”. Right: Trace link to a document based on the term “Partner”.

background color. Thus, not only the general existence of a relation between sketches is indicated, but stakeholders can also better understand the reason for its existence and explicitly see the involved sketch elements. As discussed in section I, these implicit relationships are often missed by designers, but awareness of them may possibly be beneficial for their current task.

C. Identification of Inconsistencies and Contradictions

Based on the created trace links, the AugIR's control software continuously performs a so-called impact analysis, i.e. analyzing the stakeholders' changes, detecting possible inconsistencies, and reacting accordingly. Generally, impact analysis is a non-trivial task, especially for complex changes and large software projects [33], [40].

In the AugIR, a proper impact analysis is especially important and even more challenging, because not only do we offer various simultaneous views on the project that require consistency, but as described in Sect. III, the annotations convey additional meta-information that can become inconsistent or lead to contradictions quickly. To counter these issues, the AugIR's extensive impact and relationship analysis strives to assist the stakeholders in several ways:

To identify possible inconsistencies and contradictions, the AugIR monitors the stakeholders' annotation activities. Whenever an annotation is applied to a sketch element, the AugIR identifies related sketches, i.e. sketches that also contain this element. It tests if the used annotation is also applicable to the regarded element within the related sketch(es) or if it contradicts any of the annotations that are already used 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 these transferred annotations *indirect annotations*. Indirect annotations are visualized in a slightly grayed out color to indicate that they originate from a related sketch (see Fig. 3). In addition to countering potential inconsistencies, they also provide stakeholders with a beneficial overview of all annotations that have been set for a specific element in any sketch. This is very valuable since sketches in the AugIR are usually created and annotated by multiple different stakeholders in the course of several workshops.

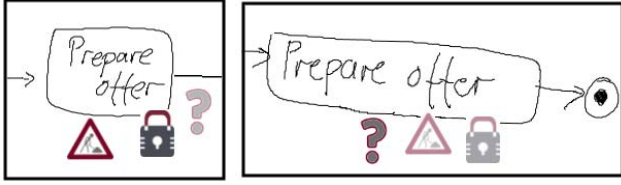


Figure 3. Indirect annotations are visualized in a slightly grayed out color. The “Revision necessary” and “Security” annotations have been placed in the left sketch, while the “Uncertainty” annotation has been placed in the right sketch.

If an annotation would lead to a contradiction, the AugIR presents a warning and advises the stakeholders to either not use that specific annotation for this element or to modify the conflicting locations accordingly so that the annotation becomes applicable. However, the stakeholders may ignore these warnings and apply the annotations anyway, because the AugIR does not restrict or hinder the stakeholders’ design and modeling activities. Instead, it strives to support the stakeholders in their collaborative work by uncovering potential issues that are easily overlooked otherwise.

V. PROTOTYPE AND VALIDATION

Figure 1 shows the AugIR prototype in which we are implementing the techniques described in this paper. The AugIR’s control software is developed using C# and Windows Presentation Foundation (WPF). We use SMART Boards [37] for the interactive displays, because such boards were already deployed successfully in similar projects [11], [42]. The hand-writing recognition is implemented by using the Microsoft.Ink libraries.

Though we are still working on the previously presented traceability techniques, a first AugIR prototype has been used by a German consulting company for internal projects in small teams up to five people over a period of three months. Although no extensive formal validation took place yet, the initial feedback was very positive. Especially the easy vertical navigation between artifacts and the intuitive refinement of sketches have been rated as very beneficial by the stakeholders.

More formalized and extensive evaluations are planned for the near future. We will begin with multiple quantitative evaluations in isolated experiments to analyze the quality of the automatically created trace links. The focus will be on finding optimal values for the thresholds μ and ν in order to achieve best recovery results while simultaneously minimizing false positive trace links.

These experiments will be followed by several case studies, using scenarios that have already been successfully used to evaluate similar approaches, like the collaborative design of a restaurant and an educational traffic simulator [26].

The validation will be concluded with qualitative evaluations in further industry projects. The usefulness of non-augmented Interaction Rooms, i.e. Interaction Rooms that use physical whiteboards and magnetic annotations, has already been shown in several real projects [3], [4], [14]. As soon

as the AugIR reaches a sufficient level of maturity, 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 effectiveness and practical benefits of our approach in general and the concepts presented above in particular.

VI. CONCLUSION AND FUTURE WORK

In this paper, we presented the Augmented Interaction Room (AugIR) and elaborated how a combination of Latent Semantic Indexing and a modified Levenshtein distance can be applied to create usable relationships between hand-drawn diagram sketches. We discussed how our approach can be beneficial for the stakeholders’ collaboration and their early design and decision making activities, and how it provides intuitive visualization and navigation as well as support for uncovering potential inconsistencies and contradictions in the created sketches.

Further research is required to determine whether automatically creating most of the trace links in the background (without requesting additional user input), or presenting suggestions to the users (and letting them decide which relations to create), is better suited for our approach. Evaluating the recovered trace links in further studies and analyzing how we can achieve the best results while simultaneously minimizing the creation of false positive trace links is a major focus of our ongoing research.

The use of wall-sized touchscreens is only a first step to augment non-digital Interaction Rooms. The integration of further devices such as cameras, voice control or special augmented-reality glasses is conceivable and would seem beneficial. Similar approaches already exist that promote for example the use of eye tracking to improve trace link recovery and maintenance [35]. Such techniques could easily be adopted for the Augmented Interaction Room and promise to further enhance its capabilities and usefulness.

REFERENCES

- [1] G. Antoniol, G. Canfora, G. Casazza, A. De Lucia, and E. Merlo, “Recovering traceability links between code and documentation,” *IEEE Trans. Softw. Eng.*, vol. 28, no. 10, pp. 970–983, Oct. 2002.
- [2] H. U. Asuncion, A. U. Asuncion, and R. N. Taylor, “Software traceability with topic modeling,” in *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1*, ser. ICSE ’10. New York, NY, USA: ACM, 2010, pp. 95–104.
- [3] M. Book, S. Grapenthin, and V. Gruhn, “Risk-aware migration of legacy data structures,” in *Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference on*, 2013, pp. 53–56.
- [4] —, “Value-based migration of legacy data structures,” in *Software Quality. Model-Based Approaches for Advanced Software and Systems Engineering*, ser. Lecture Notes in Business Information Processing, D. Winkler, S. Biffl, and J. Bergsmann, Eds. Springer International Publishing, 2014, vol. 166, pp. 115–134.
- [5] Q. Chen, J. Grundy, and J. Hosking, “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, pp. 219–226.
- [6] —, “Sumlow: early design-stage sketching of uml diagrams on an e-whiteboard,” *Software - Practice & Experience*, vol. 38, no. 9, pp. 961–994, Jul. 2008. [Online]. Available: <http://dx.doi.org/10.1002/spe.v38:9>

- [7] X. Chen and J. Grundy, "Improving automated documentation to code traceability by combining retrieval techniques," in *Proc. 2011 26th IEEE/ACM Intl. Conf. Automated Software Engineering*, ser. ASE '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 223–232.
- [8] M. Cherubini, G. Venolia, R. DeLine, and A. J. Ko, "Let's go to the whiteboard: how and why software developers use drawings," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '07. New York, NY, USA: ACM, 2007, pp. 557–566.
- [9] J. Cleland-Huang, O. Gotel, and A. Zisman, *Software and Systems Traceability*. Springer, 2012.
- [10] L. M. Covi, J. S. Olson, E. Rocco, W. J. Miller, and P. Allie, "A room of your own: What do we learn about support of teamwork from assessing teams in dedicated project rooms," in *Proceedings of Cooperative Buildings: Integrating Information, Organization and Architecture: First International Workshop, Co'Build '98*. ACM Press, 1998.
- [11] C. H. Damm, K. M. Hansen, and M. Thomsen, "Tool support for cooperative object-oriented design: gesture based modelling on an electronic whiteboard," in *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, ser. CHI '00. New York, NY, USA: ACM, 2000, pp. 518–525.
- [12] 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.
- [13] E. S. Ferguson, *Engineering and the Mind's Eye*. MIT Press, 1994.
- [14] S. Grapenthin, M. Book, V. Gruhn, C. Schneider, and K. Völker, "Reducing complexity using an interaction room: An experience report," in *Proc. 31st ACM Intl. Conf. Design of Communication*, ser. SIGDOC '13. New York, NY, USA: ACM, 2013, pp. 71–76.
- [15] J. Grundy and J. Hosking, "Supporting generic sketching-based input of diagrams in a domain-specific visual language meta-tool," in *Proceedings of the 29th international conference on Software Engineering*, ser. ICSE '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 282–291.
- [16] M. Haller, J. Leitner, T. Seifried, J. R. Wallace, S. D. Scott, C. Richter, P. Brandl, A. Gokceazade, and S. Hunter, "The nice discussion room: Integrating paper and digital media to support co-located group meetings," in *Proc. SIGCHI Conf. Human Factors in Computing Systems*, ser. CHI '10. New York, NY, USA: ACM, 2010, pp. 609–618.
- [17] R. Hebig, A. Seibel, and H. Giese, "On the unification of megamodels," in *Proceedings of the 4th International Workshop on Multi-Paradigm Modeling (MPM 2010)*, ser. Electronic Communications of the EASST, V. Amaral, H. Vangheluwe, C. Hardebolle, L. Lengyel, T. Magaria, J. Padberg, and G. Taentzer, Eds., vol. 42, 0 2011.
- [18] M. Kleffmann, M. Book, and V. Gruhn, "Towards recovering and maintaining trace links for model sketches across interactive displays," in *Traceability in Emerging Forms of Software Engineering (TEFSE), 2013 International Workshop on*, May 2013, pp. 23–29.
- [19] —, "Navigation among model sketches on large interactive displays," in *Enterprise Distributed Object Computing Conference Workshops and Demonstrations (EDOCW), 2014 IEEE 18th International*, Sept 2014, pp. 191–200.
- [20] —, "Supporting collaboration of heterogeneous teams in an augmented team room," in *Proceedings of the 6th International Workshop on Social Software Engineering*, ser. SSE 2014. New York, NY, USA: ACM, 2014, pp. 9–16.
- [21] M. Kleffmann, M. Book, E. Heibisch, and V. Gruhn, "Automated versioning and temporal navigation for model sketches on large interactive displays," in *Proc. 29th Annual ACM Symp. Applied Computing*, ser. SAC '14. New York, NY, USA: ACM, 2014, pp. 161–168.
- [22] M. B. Kokare and M. S. Shirdhonkar, "Document image retrieval: An overview," *International Journal of Computer Applications (0975 - 8887)*, vol. 1, pp. 114 – 119, 2010.
- [23] J. H. Larkin and H. A. Simon, "Why a diagram is (sometimes) worth ten thousand words," *Cognitive Science*, vol. 11, no. 1, pp. 65 – 100, 1987.
- [24] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet Physics-Doklady*, vol. 10, no. 8, 1966, pp. 707–710.
- [25] N. Mangano, A. Baker, M. Dempsey, E. Navarro, and A. van der Hoek, "Software design sketching with calico," in *Proc. IEEE/ACM Intl. Conf. on Automated Software Engineering*, ser. ASE '10. New York, NY, USA: ACM, 2010, pp. 23–32.
- [26] N. Mangano and A. Hoek, "The design and evaluation of a tool to support software designers at the whiteboard," *Automated Software Engineering*, vol. 19, no. 4, pp. 381–421, 2012.
- [27] A. Marcus, J. I. Maletic, and A. Sergeyev, "Recovery of traceability links between software documentation and source code," *International Journal of Software Engineering and Knowledge Engineering*, vol. 15, pp. 811–836, 2005.
- [28] A. Marcus, X. Xie, and D. Poshyanyk, "When and how to visualize traceability links?" in *Proceedings of the 3rd international workshop on Traceability in emerging forms of software engineering*, ser. TEFSE '05. New York, NY, USA: ACM, 2005, pp. 56–61.
- [29] G. Mark, "Extreme collaboration," *Commun. ACM*, vol. 45, no. 6, pp. 89–93, Jun. 2002.
- [30] B. Myers, S. Y. Park, Y. Nakano, G. Mueller, and A. Ko, "How designers design and program interactive behaviors," in *Visual Languages and Human-Centric Computing, 2008. VL/HCC 2008. IEEE Symposium on*, Sept 2008, pp. 177–184.
- [31] M. Petre, "Insights from expert software design practice," in *Proc. 7th Joint Meeting of the European Software Engineering Conf. and the ACM SIGSOFT Symp. Foundations of Software Engineering*, ser. ESEC/FSE '09. New York, NY, USA: ACM, 2009, pp. 233–242.
- [32] D. Pye, *The Nature and Aesthetics of Design*. A&C Black, 2000.
- [33] P. Sapna and H. Mohanty, "Ensuring consistency in relational repository of uml models," in *Information Technology, (ICIT 2007). 10th International Conference on*, dec. 2007, pp. 217 –222.
- [34] M. Schütze, P. Sachse, and A. Römer, "Support value of sketching in the design process," *Research in Engineering Design*, vol. 14, pp. 88–97, 2003.
- [35] B. Sharif and H. Kagdi, "On the use of eye tracking in software traceability," in *Proceedings of the 6th International Workshop on Traceability in Emerging Forms of Software Engineering*, ser. TEFSE '11. New York, NY, USA: ACM, 2011, pp. 67–70.
- [36] F. M. Shipman and C. C. Marshall, "Formality considered harmful: Experiences, emerging themes, and directions on the use of formal representations in interactive systems," *Computer Supported Cooperative Work (CSCW)*, vol. 8, no. 4, pp. 333–352, 1999.
- [37] Smart Technologies, "Smart board," <http://smarttech.com/smartboard>.
- [38] R. W. Soukoreff and I. S. MacKenzie, "Measuring errors in text entry tasks: An application of the levenshtein string distance statistic," in *CHI '01 Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA '01. New York, NY, USA: ACM, 2001, pp. 319–320.
- [39] G. Spanoudakis, "Plausible and adaptive requirement traceability structures," in *Proceedings of the 14th international conference on Software engineering and knowledge engineering*, ser. SEKE '02. New York, NY, USA: ACM, 2002, pp. 135–142.
- [40] G. Spanoudakis and A. Zisman, "Inconsistency management in software engineering: Survey and open research issues," in *Handbook of Software Engineering and Knowledge Engineering*. World Scientific, 2001, pp. 329–380.
- [41] N. Streitz, T. Prante, C. Müller-Tomfelde, P. Tandler, and C. Magerkurth, "Roomware: The second generation," in *CHI '02 Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA '02. New York, NY, USA: ACM, 2002, pp. 506–507.
- [42] S. Teasley, L. Covi, M. S. Krishnan, and J. S. Olson, "How does radical collocation help a team succeed?" in *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, ser. CSCW '00. New York, NY, USA: ACM, 2000, pp. 339–346.
- [43] B. Tversky, M. Suwa, M. Agrawala, J. Heiser, C. Stolte, P. Hanrahan, D. Phan, J. Klingner, M.-P. Daniel, P. Lee, and J. Haymaker, *Human Behaviour in Design: Individuals, Teams, Tools*. Springer, 2003, ch. Sketches for Design and Design of Sketches, pp. 79–86.
- [44] R. A. Wagner and M. J. Fischer, "The string-to-string correction problem," *J. ACM*, vol. 21, no. 1, pp. 168–173, Jan. 1974.
- [45] D. Wüest, N. Seyff, and M. Glinz, "Flexisketch: A mobile sketching tool for software modeling," in *Mobile Computing, Applications, and Services*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, D. Uhler, K. Mehta, and J. Wong, Eds. Springer Berlin Heidelberg, 2013, vol. 110, pp. 225–244.
- [46] C. Zannier and F. Maurer, "Comparing decision making in agile and non-agile software organizations," in *Proceedings of the 8th International Conference on Agile Processes in Software Engineering and Extreme Programming*, ser. XP'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 1–8.