

Supporting Collaboration of Heterogeneous Teams in an Augmented Team Room

Markus Kleffmann
University of Duisburg-Essen
Essen, Germany
markus.kleffmann
@paluno.uni-due.de

Matthias Book
University of Duisburg-Essen
Essen, Germany
matthias.book
@paluno.uni-due.de

Volker Gruhn
University of Duisburg-Essen
Essen, Germany
volker.gruhn
@paluno.uni-due.de

ABSTRACT

It is often difficult for a team of stakeholders with heterogeneous backgrounds to maintain a common understanding of a system's structure and the challenges in its implementation. Thus, especially in complex software projects, risks and inconsistencies are easily overlooked. In this paper, we present the concept of an Augmented Interaction Room (AugIR), i.e. a physical team room whose walls are outfitted with wall-sized touchscreens that visualize different aspects of a software system in the form of various model sketches. These sketches can be annotated by the stakeholders in order to explicitly mark important elements or indicate aspects that are critical for project success. The AugIR strives to support the collaborative work of heterogeneous teams and especially targets the inclusion of non-technical stakeholders into the communication process. Therefore, the AugIR continuously monitors the stakeholders' design and modeling activities and analyzes the relationships between annotated contents to automatically uncover inconsistencies, contradictions and hidden potential project risks.

Categories and Subject Descriptors

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

General Terms

Design, Human Factors

Keywords

Cooperative Design; Electronic Whiteboards; Sketches; Object-oriented Modeling

1. INTRODUCTION

It is often difficult for team members with heterogeneous backgrounds to maintain a common understanding of a com-

plex software project. Common process models such as Scrum generally only provide the organizational framework for the development process, but do not offer effective support for the goal-oriented work on the project's challenges. We therefore introduced the idea of an *Interaction Room (IR)* in a previous paper [9]. The IR is a persistent physical team room that helps stakeholders to foster a common understanding of the project's most important risk and value drivers. The walls of the room are outfitted with diagrams and sketches, such as process models, class diagrams or migration maps. Stakeholders can annotate them with markers, so-called *Annotations*, which are little magnetic pins with imprinted graphics that have specific semantics. 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.

The IR relies solely on a physical visualization approach with whiteboards and magnetic annotations. While our experiences in industry projects have shown the usefulness of this room for supporting the collaboration of heterogeneous teams in early design and decision making phases of complex software development projects [8, 10, 22], the physical presentation has some major disadvantages [12, 19, 20]:

First and foremost, sketches and drawings on different boards can quickly become inconsistent. This issue does not only apply to the diagram contents but also to the used annotations which are placed by individual team members and are therefore often subjective. Aspects that are important or uncertain to some stakeholders may seem unimportant or fully clear to others.

Furthermore, annotations can indicate potential project risks such as dependencies to insufficiently understood elements. Uncovering those hidden risks is often difficult when working with a pure physical presentation, as no explicit relationships or dependencies exist between the physical artifacts.

In addition to that, the size of the diagrams and sketches is limited to the available physical drawing area. Modeling can be very laborious because conceptually trivial changes such as the simple movement of an element may force the users to erase and completely redraw large parts of the diagrams. Furthermore, there is no active support for drawing or modeling, like auto-completion or straightening of lines. Archiving or digitizing the annotated sketches can often be time-consuming as well.

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 ACM 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.

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

SSE'14, November 17, 2014, Hong Kong, China
ACM 978-1-4503-3227-9/14/11
<http://dx.doi.org/10.1145/2661685.2661688>

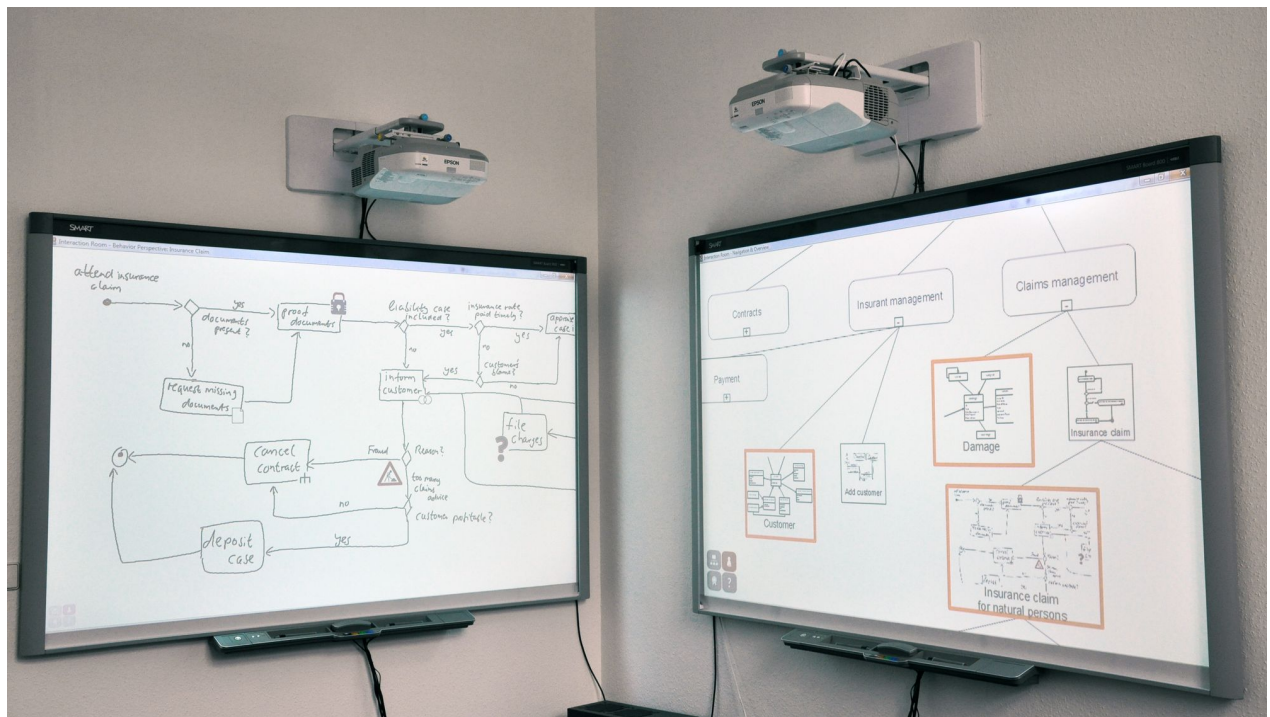


Figure 1: Prototype of the Augmented Interaction Room (AugIR).

Our contribution in this paper is the proposition of a technical augmentation of the Interaction Room that counters the previously described issues: We introduce the *Augmented Interaction Room (AugIR)*, which is a physical room whose walls are outfitted with large wall-mounted touchscreens (see Fig. 1). Generally, studies have shown that the usage of large displays can be beneficial for understanding complex systems through improved usage of spatial information to convey meaning and significance [1], to create a more immersive experience while working [7], and to let people take in more information at once [3]. The physical movement involved when working with large displays is generally regarded positively by most users and often improves their performance [5, 6].

The AugIR provides teams with a dedicated project space that supports effective communication as well as collaborative system design and implementation. Each display is dedicated to a particular modeling perspective, as shown in Fig. 2. In the course of their discussions, stakeholders can sketch models (e.g. business process models, class diagrams etc.) directly on these touch screens and annotate them with digital markers that explicitly highlight particularly valuable, complex, risky or insufficiently understood components. In this paper, we will focus on these annotations and how they can help stakeholders to uncover inconsistencies, contradictions and hidden potential project risks, as the AugIR automatically monitors the stakeholders’ design and modeling activities and analyzes the relationships between annotated elements.

We begin with an overview of related work in section 2, followed by a problem statement and motivation in section 3. Section 4 briefly describes the general Interaction Room design concepts, while section 4.1 summarily explains the creation of sketch relationships that are extensively used in

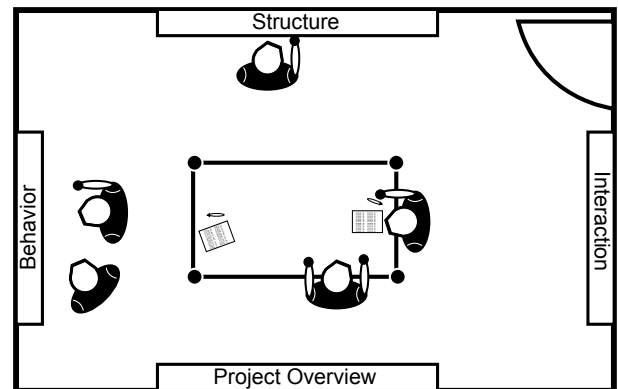


Figure 2: Example arrangement of perspectives in a four-wall AugIR layout.

our approach. In section 4.2, we discuss how annotations in the AugIR can be used to automatically identify inconsistencies and contradictions, and section 4.3 describes how potential project risks may be derived from annotated elements and their relationships. In section 5, 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 6.

2. RELATED WORK

Several companies and research centers are experimenting with collocating teams in rooms, so-called “war rooms” or project rooms [18, 35]. Studies with software developers have shown an increased productivity and personal satisfac-

tion when all project members work side-by-side in these team rooms [44]. In almost any scenario, whiteboards and flip charts are frequently used for collaborative design activities [16], visualization of important project knowledge [18], as well as to help stakeholders to monitor the progress and focus on the critical aspects of a project [9]. As already motivated in section 1, replacing physical whiteboards with large interactive displays can be very beneficial, especially in these team rooms.

Streitz et al. [43] present Roomware, which integrates information and communication technology in various room elements like tables, chairs, and walls. By equipping the environment with different interfaces and network sensors, Roomware offers new ways of interacting with information by making communication and information access ubiquitously available.

Another approach is presented by Haller et al. [24]. They introduce the so-called NiCE Discussion Room, which is a technically augmented meeting room that especially focuses on the integration of paper artifacts. The room is outfitted with a large display for collaborative work, onto which handwritten paper notes can be transferred via a special interface.

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.

A promising approach is presented by Chen, Grundy and Hosking [12, 13]. It introduces 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. Based on this work, Grundy and Hosking have developed a meta-tool called Marama for creating domain-specific visual language tools that run as Eclipse plug-ins [23].

Mangano et al. [33] present Calico, a sketching tool that is especially suited for modeling during the early phases of a software development project. It allows the designer to create free-hand sketches on electronic whiteboards or tablets. By circumscribing an area on the board, the designer can create a so-called “scrap”, which gives certain regions within the sketch an informal importance. The user can move the scraps and their content around, as well as combine and connect them to create relationships between them.

Hammond and Davis present Tahuti [25], which allows the user to draw diagrams on large multi-touch displays. The user can either choose to display all elements as they have been drawn in free-hand mode, or they can be interpreted as UML elements and replaced by the corresponding symbols. This replacement uses pattern recognition in multiple steps. The approach combines the simplicity of a paint tool with the editing support of a UML editor.

Damm, Hansen and Thomsen [19] observed software engineers during their work and derived important design criteria for object-oriented modeling tools. Based on their observations, they present Knight, a tool that is similar to Tahuti. Knight offers a free-hand mode, in which all sketches appear on the wall exactly as they have been drawn, as well as a UML mode, in which all drawn lines and shapes are directly replaced with UML elements. The drawn diagrams may be incomplete and do not have to follow strict UML semantics.

Glinz et al. [45] present a mobile application called FlexiSketch. They argue that most software engineers prefer paper and pencil when sketching ideas and performing modeling activities, especially when working in the field, e.g. gathering information from stakeholders. Therefore, FlexiSketch is a sketching tool for free-hand modeling, developed for the use on mobile devices. Users can draw informal sketches, enrich them with annotations, and later transform them into semi-formal models for further usage.

Ansloew et al. [2] argue that most software projects are developed by teams of people, yet most visualization and modeling tools are primarily designed to support only a single user. Therefore, they present SourceVis, a tool for large interactive multi-touch surfaces. SourceVis is not a sketch tool but focuses on software visualization during collaboration by supporting the analysis of the structure, evolution and vocabulary of software systems.






3. MOTIVATION FOR DIGITAL ANNOTATIONS AND AUTOMATION

In the Interaction Room, stakeholders can annotate diagrams and sketches with a variety of graphical annotations [9, 10]. In the non-augmented IR, i.e. the Interaction Room that uses physical whiteboards instead of electronic whiteboards, annotations are realized as magnetic pins with imprinted graphics. They can be placed anywhere within the diagram sketches and are used 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 of important aspects.

The Interaction Room method defines a total number of 23 different annotations that can be used within the model sketches. Each annotation has its unique symbol and a specific semantic. Table 1 shows five examples of frequently used annotations: The “*Frequent Changes*” annotation denotes elements that are often modified and change regularly. 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 individual or group uncertainty, e.g. if a process is not yet well understood or insufficiently specified. The “*Security*” annotation denotes elements that bear a high need for security, e.g. components that access sensitive user data or passwords.

Although the usefulness of the non-augmented Interaction Room in general and the annotations in particular have already been shown in previous industry projects [8, 10, 22], the use of pure physical annotations has some issues and can greatly benefit from digitization and automation. We will illustrate these issues with simple examples that use the annotations introduced above:

Table 1: Examples of Interaction Room annotations.

Symbol	Meaning
	Frequent Changes
	Revision Necessary
	Immutable
	Uncertainty
	Security

- Using annotations on physical whiteboards can quickly lead to inconsistencies between different (annotated) model sketches. For example, assuming C is a class that occurs in two diagram sketches S_1 and S_2 , and C is annotated as “Immutable” in S_1 using the stop sign annotation. To avoid an inconsistency, C must also be marked as immutable in the context of S_2 . Otherwise, the stakeholders would be able to modify C in S_2 , e.g. by changing a method signature, which would violate the annotation that has been used in S_1 .

Inconsistencies can arise due to various reasons:

First of all, there exist no explicit relationships between the model sketches. Therefore, annotating an element in one sketch does not automatically add the same annotation to all occurrences of this element in other sketches. Since complex software projects usually consist of a multitude of different models and sketches that have been created by different teams of stakeholders, searching for all occurrences of an annotated element, especially across physically drawn artifacts, can be very laborious.

Furthermore, annotations are placed by individual team members and are often subjective. Aspects that are important or uncertain to some stakeholders may seem unimportant or fully clear to others. Different team compositions often result in differently annotated sketches. This can lead to inconsistencies, since different diagrams in software projects are usually modeled by different groups of stakeholders and therefore the same element may be annotated differently in different sketches.

Checking the consistency for annotated elements is laborious and error-prone when using physical whiteboards. The effort for a consistency check increases exponentially with every new sketch that is added to the repository, because every artifact has to be manually checked for consistency with every other artifact whenever an element is annotated.

- Furthermore, 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 artifact, uncovering them when working with different artifacts is much more difficult, especially when using physical whiteboards where no explicit relationships exist between the artifacts.

For example, assuming E is a process element that uses a certain interface I , and E is annotated as “Immutable”. Annotating the interface I with the “Revision Necessary” annotation could indicate a potential contradiction, because changing the interface would probably require changing the immutable process element that uses it.

Since this must not always be the case, it remains the stakeholders’ responsibility to decide if an annotation is suitable in a specific scenario. However, it is important to notify the stakeholders of such potential contradictions, as they can be easily overlooked in a complex software project, especially when sketches are created by different heterogeneous teams.

Similar to the consistency check described above, manually checking the sketches for potential contradictions is very time-consuming when working with physical artifacts.

- Lastly, annotations may indicate potential project risks. For example, assuming P is a process that contains many elements which are marked with the “Uncertainty” annotation. Using P as a sub-process in many other business processes indicates a potential project risk, since these processes would depend on or use a process that contains much uncertainty and is probably insufficiently understood by the stakeholders.

Since there exist no explicit relationships between artifacts when using physical whiteboards, an analysis of such dependencies and an aggregation of used annotations (e.g. the total number of occurrences of the “Uncertainty” annotation in a specific process model) is laborious and error-prone. Therefore, potential project risks that are indicated by the used annotations and the relationships between annotated elements could remain hidden and are easily overlooked.

In the following sections, we will discuss how the AugIR strives to support heterogeneous stakeholder teams and to counter the issues described above by employing digitized annotations and automated identification of inconsistencies, contradictions and potential projects risks.

4. AUGMENTED INTERACTION ROOM

As already introduced in section 1, the Augmented Interaction Room (AugIR) is a physical team room whose walls are outfitted with large wall-mounted touchscreens (see Fig. 1). Diagrams and sketches are directly drawn on 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 free-hand drawing and writing is much more efficient for model sketching, and that informal notations are preferred by most users over formal graphical modeling languages [16, 20]. 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 [9]. Furthermore, an easy and intuitive handling and input is especially important in early design and modeling phases, when designers often use low-detail models and require the possibility to work fast enough in order keep up with their thought process [21, 36]. Nevertheless, the AugIR offers significant modeling and drawing support: If enabled by the stakeholders, hand-drawn elements and words are automatically replaced with optimized computer-generated symbols and text using pattern and handwriting recognition. Still, all sketches can remain incomplete or even inconsistent, as the AugIR does not enforce the use of a strict formal modeling language.

While stakeholders in the non-augmented Interaction Room use magnetic markers to pin annotations onto the whiteboards, the AugIR requires a different approach: On any display, a bar with icons can be opened that shows all annotations that are available in the current context. Drawing an arbitrary imaginary symbol onto an annotation assigns this annotation to that symbol. When this symbol is drawn onto or next to a sketch element, the AugIR automatically replaces it with the assigned annotation using pattern recognition. Thus, each stakeholder can define his own personal set of symbols and assigned annotations, which makes the annotation process easy to understand and intuitive to perform, especially for non-technical stakeholders.

While other approaches described in section 2 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 [23].

Each display shows a certain perspective onto the system (see Fig. 2). The perspective of each display determines the kind of content that can be displayed on it. At any time, the stakeholders can change the perspective on any display dynamically and the perspectives may even switch automatically, e.g. when the stakeholders navigate within the project and open a new diagram type. Being able to dynamically change the wall perspectives on each display is very important, because during their design activities users often switch focus and work on different diagram types [34]. Having a fixed position for each perspective within the room would require the stakeholders to move around and switch between the boards frequently, e.g. when exploring the project, which would make the design process unnecessarily laborious.

Figure 2 shows an example of a possible four-wall AugIR layout that includes three walls for modeling the structural, behavioral and interactive aspects of the system, as well as a fourth wall that is used for navigation and overview of artifact and sketch relationships. Figure 1 shows an example from our current prototype that models the data structures

and business processes of an insurance company. The behavior perspective is displayed on the left wall and the project overview perspective is displayed on the right wall.

The following sections discuss in detail how the AugIR specifically supports the collaboration of heterogeneous teams.

4.1 Creation of Sketch Relationships

The automated identification of inconsistencies, contradictions and potential projects risks relies on the relationships and dependencies between the model sketches. These so-called trace links are created mostly automatically by using a combination of prospective and retrospective traceability techniques, as described in our earlier work [28].

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 [4, 17, 31]. One can effectively combine different techniques to employ the strengths of several approaches, as already demonstrated by Asuncion, Asuncion and Taylor [4], Chen, Hosking and Grundy [14, 15] and others.

Basically, if users repeatedly work with certain sketches on different walls, the AugIR uses prospective traceability techniques to create trace links between these artifacts, 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.

In addition, we apply retrospective traceability techniques by using various information retrieval techniques which mostly rely on textual information to create traceability links. Therefore, trace links are also created if artifacts share the same elements or textual contents, e.g. if the same class appears in a class diagram as well as in a sequence diagram, or if sketch elements are referenced in external artifacts, e.g. a class that is textually referenced in a bug report or specification document.

Furthermore, explicit trace links between sketches and artifacts can also be created manually by the users, to counter the issue that creation of retrospective trace links is a very difficult task for non-textual artifacts, such as screenshots [32].

As described in a previous paper [29], we store these relationships and dependencies between model sketches in a so-called mega- or macromodel, which is a graph that contains models and their relationships [26]. Therefore, the trace links can also be used for context-sensitive and intuitive gesture-based navigation [29].

4.2 Identification of Inconsistencies and Contradictions

Based on the trace links described in the previous section, the AugIR’s control software continuously performs a so-called impact analysis, i.e. analyzing the users’ changes, detecting possible inconsistencies, and reacting accordingly. When, for example, a class C is contained in two different sketches, S_1 and S_2 , and C is modified in S_1 , e.g. by renaming the class, these changes must also be reflected in S_2 . Generally, impact analysis is a non-trivial task, especially for complex changes and large software projects [39, 42].

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 the annotations convey additional meta-information that also can become inconsistent or lead to contradictions quickly. To counter the issues described in section 3, 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 an element, the AugIR uses traceability techniques to identify related artifacts, e.g. artifacts that also contain this element. It tests if the used annotation is also applicable to the regarded element within the related artifacts 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 artifacts to avoid inconsistencies. Otherwise, the AugIR presents an appropriate 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.

To uncover opposing annotations, we created the Cartesian product of all annotations and extensively analyzed all elements in the resulting set regarding to potential inconsistencies or contradictions they could possibly cause.

4.3 Identification of Potential Project Risks

In addition to the automated identification of inconsistencies and contradictions, the AugIR also strives to support the stakeholders by uncovering potential risks in their projects. Therefore, it analyzes the relationships and dependencies between annotated elements and also automatically aggregates a variety of metrics (e.g. how often which types of annotations have been used).

If, for example, a process contains many “Uncertainty” annotations, and if this process is used at many locations within the software project, a potential risk for the project success is indicated, since a central component of the system is probably insufficiently understood by the stakeholders.

Further examples for potential project risks are interfaces that are annotated with the “Frequent changes” or the “Revision necessary” annotation, because especially interfaces should be as constant as possible in large software development projects.

Another potential project risk is given, when an element that is annotated with the “Security” annotation depends on another element that is annotated with the “Uncertainty” marker, because this dependency may (probably accidentally) compromise the security requirements.

In addition to regarding the relationships and dependencies between annotated elements, potential project risks can also be derived from various metrics that are automatically aggregated from the used annotations. For example, an overall high number of used “Uncertainty” annotations across various artifacts may indicate a potential project risk

since it suggests that many parts of the software project are not well understood by certain stakeholders.

The current AugIR prototype continuously analyzes the project for potential risks and presents a list with all identified deficits after each design session. However, deciding when and how to automatically present appropriate visual warnings *during* a design session is still an open research question. On the one hand, stakeholders should be informed about such deficits as soon as possible. On the other hand, we must not disturb them in their concentration or hinder their work flow. For a convenient solution we will have to evaluate various research approaches that strive to solve similar problems like Sideshow by Cadiz et al. [11] and the OASIS framework by Iqbal et al. [27].

5. PROTOTYPE AND VALIDATION

Figure 1 shows the AugIR prototype that we are currently working on. The AugIR’s control software is developed using C# and Windows Presentation Foundation (WPF). We use SMART Boards [41] for the interactive displays, because such boards were already deployed successfully in similar projects [19, 44]. Handwriting recognition is implemented by using the Microsoft.Ink libraries together with an enhanced InkCanvas control, while pattern recognition is realized via Windows.Ink.

Although no extensive formal validation took place yet, the prototype has already been intensively tested by colleagues and students, all of whom had practical and theoretical experience in software engineering and the design of software systems. In observing them, we noticed that all users were able to use the digital whiteboards and the AugIR’s annotations intuitively after a short introduction. Nevertheless, we found some shortcomings in the current prototype: For example, we still need a way to efficiently transfer contents from one wall to another. Sketch elements can already be moved around by simple drag gestures, and can be transferred to adjacent displays by moving them to the corresponding border of the display. This works well for directly adjacent displays, but gets laborious when moving elements across several displays. For a more convenient solution, we will need to evaluate different techniques, such as hyperdragging [38] or pick-and-drop [37].

More formalized and extensive evaluations are planned for the near future. Since a validation of our approach in one single step is nearly impossible, we will begin with multiple quantitative evaluations in isolated experiments, i.e. testing and comparing different alternative visualization and interaction concepts under laboratory conditions, e.g. different undo / redo techniques. 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 [34].

The validation will be concluded with qualitative evaluations in 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 [8, 10, 22]. 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.

6. CONCLUSION AND FUTURE WORK

In this paper, we presented the Augmented Interaction Room (AugIR) and discussed how annotations can be used to explicitly indicate important elements, highlight aspects that are especially critical for project success, uncover uncertainties of individual team members and make implicit project knowledge explicit. Furthermore, we described how the AugIR strives to automatically uncover inconsistencies, contradictions and hidden potential project risks based on the used annotations and the relationships between annotated elements, and motivated how this can be beneficial for heterogeneous teams in complex software projects. By further analyzing how and when annotations are used in industry projects, we strive to derive a more extensive list of potential project risks and how they may be indicated by the use of specific annotations.

The current AugIR prototype only supports sketches that have been created directly on the walls. However, it is also conceivable that users will wish to import externally created diagrams into the room. Though we already selected the file formats with respect to that [28, 30], how we can efficiently import many externally created sketches or diagrams at once and automatically insert them into our graph at the proper location is still an open research question.

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 [40]. Such techniques could easily be adopted for the Augmented Interaction Room and promise to further enhance its capabilities and usefulness.

7. REFERENCES

- [1] C. Andrews, A. Endert, and C. North, “Space to think: large high-resolution displays for sensemaking,” in *Proc. SIGCHI Conf. on Human Factors in Computing Systems*, ser. CHI '10. ACM, 2010, pp. 55–64.
- [2] C. Anslow, S. Marshall, J. Noble, and R. Biddle, “Interactive multi-touch surfaces for software visualization,” in *Proc. Workshop on Data Exploration for Interactive Surfaces DEXIS 2011*, 2011, pp. 20–23.
- [3] C. Anslow, S. Marshall, J. Noble, E. Tempero, and R. Biddle, “User evaluation of polymetric views using a large visualization wall,” in *Proc. 5th intl. symp. on Software visualization*, ser. SOFTVIS '10. ACM, 2010, pp. 25–34.
- [4] 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. ACM, 2010, pp. 95–104.
- [5] R. Ball and C. North, “The effects of peripheral vision and physical navigation on large scale visualization,” in *Proceedings of graphics interface 2008*, ser. GI '08. Toronto, Ont., Canada, Canada: Canadian Information Processing Society, 2008, pp. 9–16.
- [6] R. Ball, C. North, and D. A. Bowman, “Move to improve: promoting physical navigation to increase user performance with large displays,” in *Proc. SIGCHI Conf. Human Factors in Computing Systems*, ser. CHI '07. ACM, 2007, pp. 191–200.
- [7] X. Bi and R. Balakrishnan, “Comparing usage of a large high-resolution display to single or dual desktop displays for daily work,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '09. ACM, 2009, pp. 1005–1014.
- [8] 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.
- [9] —, “Seeing the forest and the trees: focusing team interaction on value and effort drivers,” in *Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering*, ser. FSE '12. ACM, 2012, pp. 30:1–30:4.
- [10] —, “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. Biffi, and J. Bergsmann, Eds. Springer International Publishing, 2014, vol. 166, pp. 115–134.
- [11] J. J. Cadiz, G. Venolia, G. Jancke, and A. Gupta, “Designing and deploying an information awareness interface,” in *Proceedings of the 2002 ACM conference on Computer supported cooperative work*, ser. CSCW '02. ACM, 2002, pp. 314–323.
- [12] 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.
- [13] —, “Sumlow: early design-stage sketching of uml diagrams on an e-whiteboard,” *Software - Practice & Experience*, vol. 38, no. 9, pp. 961–994, Jul. 2008.
- [14] X. Chen and J. Grundy, “Improving automated documentation to code traceability by combining retrieval techniques,” in *Proceedings of the 2011 26th IEEE/ACM International Conference on Automated Software Engineering*, ser. ASE '11. IEEE Computer Society, 2011, pp. 223–232.
- [15] X. Chen, J. Hosking, and J. Grundy, “A combination approach for enhancing automated traceability (nier track),” in *Proceedings of the 33rd International Conference on Software Engineering*, ser. ICSE '11. ACM, 2011, pp. 912–915.
- [16] 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. ACM, 2007, pp. 557–566.
- [17] J. Cleland-Huang, O. Gotel, and A. Zisman, *Software and Systems Traceability*. Springer, 2012.
- [18] 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.

- [19] 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. ACM, 2000, pp. 518–525.
- [20] 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.
- [21] E. S. Ferguson, *Engineering and the Mind's Eye*. MIT Press, 1994.
- [22] 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. ACM, 2013, pp. 71–76.
- [23] 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. IEEE Computer Society, 2007, pp. 282–291.
- [24] M. Haller, J. Leitner, T. Seifried, J. R. Wallace, S. D. Scott, C. Richter, P. Brandl, A. Gokcezade, and S. Hunter, "The nice discussion room: Integrating paper and digital media to support co-located group meetings," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '10. ACM, 2010, pp. 609–618.
- [25] T. Hammond and R. Davis, "Tahuti: a geometrical sketch recognition system for uml class diagrams," in *ACM SIGGRAPH 2006 Courses*, ser. SIGGRAPH '06. ACM, 2006.
- [26] 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, vol. 42, 0 2011.
- [27] S. T. Iqbal and B. P. Bailey, "Oasis: A framework for linking notification delivery to the perceptual structure of goal-directed tasks," *ACM Trans. Comput.-Hum. Interact.*, vol. 17, no. 4, pp. 15:1–15:28, Dec. 2010.
- [28] 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*, 2013, pp. 23–29.
- [29] —, "Navigation among model sketches on large interactive displays," in *International Workshop on Methodical Development of Modeling Tools (ModTools14)*, 2014, to appear.
- [30] M. Kleffmann, M. Book, E. Hebisch, and V. Gruhn, "Automated versioning and temporal navigation for model sketches on large interactive displays," in *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, ser. SAC '14. ACM, 2014, pp. 161–168.
- [31] S. Klock, M. Gethers, B. Dit, and D. Poshyvanyk, "Traceclipse: an eclipse plug-in for traceability link recovery and management," in *Proceedings of the 6th International Workshop on Traceability in Emerging Forms of Software Engineering*, ser. TEFSE '11. ACM, 2011, pp. 24–30.
- [32] 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.
- [33] N. Mangano, A. Baker, M. Dempsey, E. Navarro, and A. van der Hoek, "Software design sketching with calico," in *Proceedings of the IEEE/ACM international conference on Automated software engineering*, ser. ASE '10. ACM, 2010, pp. 23–32.
- [34] 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.
- [35] G. Mark, "Extreme collaboration," *Commun. ACM*, vol. 45, no. 6, pp. 89–93, Jun. 2002.
- [36] D. Pye, *The Nature and Aesthetics of Design*. A&C Black, 2000.
- [37] J. Rekimoto, "Pick-and-drop: A direct manipulation technique for multiple computer environments," in *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '97. ACM, 1997, pp. 31–39.
- [38] J. Rekimoto and M. Saitoh, "Augmented surfaces: A spatially continuous work space for hybrid computing environments," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '99. ACM, 1999, pp. 378–385.
- [39] 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.
- [40] 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. ACM, 2011, pp. 67–70.
- [41] Smart Technologies, "Smart board," <http://smarttech.com/smartboard>.
- [42] 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.
- [43] 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. ACM, 2002, pp. 506–507.
- [44] 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. ACM, 2000, pp. 339–346.
- [45] D. Wüest, N. Seyff, and M. Glinz, "Flexisketch: A mobile sketching tool for software modeling," in *Mobile Computing, Applications, and Services*. Springer Berlin Heidelberg, 2013, vol. 110, pp. 225–244.