Are Smartphones Better Than CRC Cards?

Rainer Lutz
Computer Science Department
University of Trier
Trier, Germany
lutzr@uni-trier.de

Sascha Schäfer
Computer Science Department
University of Trier
Trier, Germany
s4sascha@uni-trier.de

Stephan Diehl
Computer Science Department
University of Trier
Trier, Germany
diehl@uni-trier.de

ABSTRACT

During early phases of a software development process co-located group work is an important technique that involves all stakeholders to derive requirements and a design of a future software system. However, such group work is usually applied without any computer-assistance and often faces the problem that information is not well preserved for subsequent steps. In order to examine whether group work benefits from computer-assistance, we developed CREWSpace. It leverages mobile devices to allow simultaneous interaction with a shared software model. In particular, it implements a digital variant of the CRC method. In this paper, we discuss advantages and disadvantages of the traditional CRC method, briefly introduce CREWSpace along with important implementation details, and focus on a qualitative usability study. Its results suggest that our prototype keeps the advantages of traditional CRC method and compensates many of its weaknesses.

Categories and Subject Descriptors

D.2.1 [**Software Eng.**]: Requirements/Specifications—*Tools*

Keywords

requirements engineering, computer-supported collaborative work, tools and environments, crc cards, mobile devices

1. INTRODUCTION

While the answer to the question raised in the title is obvious when it comes to making a call, it is less obvious when it comes to collaboratively analyzing requirements.

Co-located group work is an important method to elicit requirements and develop a fundamental model of a future software system. People gather in the same room, provide ideas, discuss their advantages and disadvantages, and finally, produce a first version of the system to be built. However, these meetings are usually held without any computer-assistance and often face the problem that particular information is not well preserved for subsequent steps.

On the other hand, a lot of activities in people's everyday life have already been shifted to the digital world. Due to the prevalence of mobile devices nowadays, smartphones and tablet computers are not only used for communication, but also to augment physical activities or replace low-tech artifacts and even improve on them. For instance, hiking has been augmented into geocaching and paper calendars have been widely replaced by electronic ones. Thus, it seems promising to leverage such devices to augment activities and replace real-world artifacts during the software development process, as well. Moreover, mobile devices can help to eliminate media breaks as all data must simply be available in a digital form in order to be processed.

Furthermore, mental execution or mental walkthrough are typical analysis methods in software engineering [14, 26]. During group work activities role play can be used to analyze certain aspects of a software model. In these role plays each person typically takes on the role of one or more elements of the model (e.g. classes of the future system) and acts for them. While role play has been applied in software engineering education of most phases, in real software development, it is mainly used in the requirements phase to bridge the communication gap between developers, customers, and users.

Recently, we developed a series of prototypes [6, 7, 15] that support co-located collaboration for model analysis based on the CRC method introduced by Beck and Cunningham [3]. Our latest tool, CREWSpace [15], allows users to simultaneously interact with a shared software model using touch devices like smartphones or tablets to create their model and analyze it through role play.

In this paper, we discuss advantages and drawbacks of the traditional CRC method and briefly explain how CRC sessions with CREWSpace differ from common ones. Furthermore, we describe important aspects of the design and the implementation of our prototype. The focus of this paper, however, is on the usability study where we present in which way our participants used CREWSpace, how they perceived it, and to what extent our prototype can remedy the identified disadvantages.

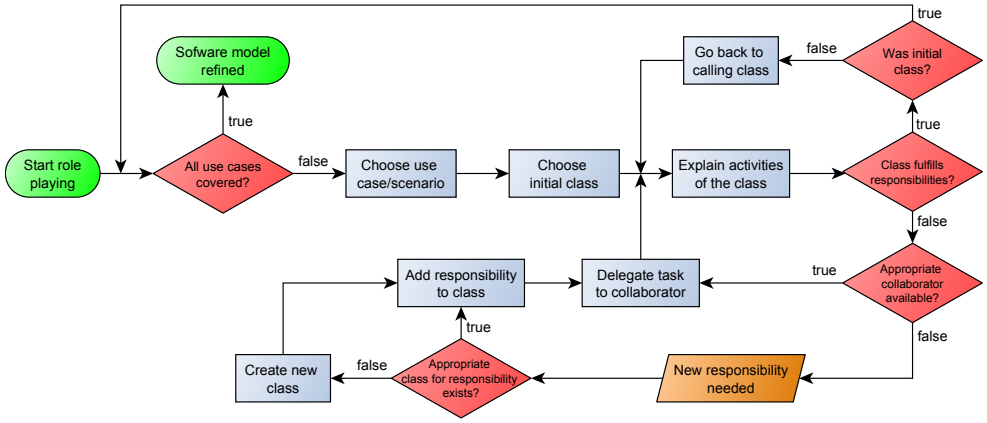
2. MODELLING WITH CRC CARDS

The CRC method has been first introduced by Beck and Cunningham at OOPSLA in 1989 [3]. In their paper they explain how index cards and role play can be used to develop an object-oriented model of a future software system. The role play itself is based on use cases [13] and use case scenarios, which describe functional requirements and capture the externally visible behavior of a system from a user's perspective. In general, the CRC method is intended to supplement the requirements analysis as requirements provided by customers are often vague or simply do not match the vocabulary of requirements engineers.

CRC cards are divided into three parts: a header for the class name and two columns; the left one for responsibilities and the right one for collaborators. Responsibilities list both the tasks a

SAC'14, March 24-28, 2014, Gyeongju, Korea.

Table 1: Process of traditional CRC sessions

P1	**Identifying use cases:** During the first phase relevant use cases, possible scenarios, and their actors are identified in a brainstorming session. The gathered information is an essential precondition for the third phase and thus may be stored in several ways, e.g., in a textual form, as a use case diagram, or a combination of both [11].
P2	**Identifying classes:** Here, the team identifies candidate classes, which serve as a starting point for the next phase. The name of each candidate class is written on a separate CRC card. Obvious responsibilities and collaborators may also be added. Missing information will be added in the third phase.
P3	**Analyzing use cases:** Finally, the team uses role play to refine the initial software model from the second phase. Therefore, each team member acts for a certain number of classes. Role play resembles a simple "what happens next" game where the team walks through a use case scenario—a possible path through a use case—and assesses whether the software model is appropriate for that particular scenario. To this end, one of the team members is said to be active and explains which responsibilities her CRC card is able to fulfill and whether there is a need for collaboration with other classes. During the role play the group may add responsibilities, search for collaborations between classes, or even create new cards in order to refine the software model. The following figure depicts the complete procedure: 
P4	**Transfer to UML:** Once all classes, their responsibilities, and collaborators have been identified, it is crucial to translate the CRC model to a more powerful modeling language like UML. In order to emphasize fundamental relationships between classes, the team members may stick the CRC cards to a whiteboard and connect them with hand-drawn edges.

class is able to fulfill and the information it contains. Collaborators are those cards a class has to communicate with in order to fulfill a certain task or to access certain data [3, 4]. During so called CRC sessions all stakeholders are involved and try to identify the central classes of the system to be built. Such groups typically consist of four to seven members with different expertise and are assisted by a moderator. A CRC session comprises four subsequent phases (P1-P4, cf. Tab. 1) and requires only pens, a set of blank CRC cards, and a table to spread them out [4].

Before we were able to develop a tool for computer-assisted CRC sessions, we had to study benefits and drawbacks of the traditional CRC method to derive requirements for such a prototype. In the following we describe all advantages (A1–A3) and disadvantages (D1–D4) relevant for computer-assisted CRC sessions along with the resulting *requirements*:

A1: First of all, the traditional CRC method does not need any kind of technical support. It is independent from operating systems, computers, and even electricity. All you need is index cards, pens, and a table. *Although this cannot be achieved with computer-assistance, a low entry barrier should be preserved.*

A2: CRC sessions are based on group work activities and role play. *Thus, our prototypes should not only support multi-user interactions, but also allow for fair collaboration between all stakeholders. Furthermore, a tool should display which components a certain user is allowed to modify or not.*

A3: According to Beck et al. one of the biggest advantages of the traditional CRC method is the tangibility of paper cards and the possibility to identify with them [3]. *Hence, a tool should preserve such tangibility as good as possible.*

D1: Paper cards are at a disadvantage when it comes to changing or removing information. Previous experience showed that it is not trivial to correctly identify names of classes and responsibilities at the beginning of an analysis, and that these names may change during the course of a CRC session [5]. For instance, if the name of a card, which serves as collaborator for other cards, is changed, a certain rewriting effort is necessary to keep all names consistent among several cards. *A computer-assisted variant should provide editable text fields as well as automatic features, e.g., for the propagation of changes.*

D2: As Ambler [2] put it, "CRC modeling leads directly into class diagramming". But when performing a traditional CRC session no digital version of the cards is maintained or produced and thus UML class diagrams have to be modeled in a separate step. *Therefore, a prototype should implement at least basic UML concepts and a simple export functionality to common UML tools.*

D3: During role play it may quickly become non-trivial to recall the current state and how it was reached. *Thus, a computer-assisted variant should preserve and visualize such a call history, provide methods to navigate through it, and allow to restart the role play from any point in time.*

D4: Managing different documents (requirements, use case diagrams, etc.) in addition to a set of CRC cards might distract the team members from their actual task. *Hence, a tool should maintain digital copies of all those documents and provide individual and fast access to them.*

3. COMPUTER-ASSISTED CRC SESSIONS

This section briefly explains how CRC sessions (Fig. 1) with CREWSpace are conducted and contrasts it with the traditional method. Please note that our previous work [15] provides a detailed description of the entire process.

Assuming that a computer maintaining a shared workspace in the main application and Android-enabled mobile devices, which serve as controls and as private workspaces for the team members, are available, computer-assisted CRC sessions are actually very similar to traditional ones. First, use cases are identified, then an initial software model is developed and further refined by using role play for use case analysis as described in Tab.1. Instead of writing paper cards, digital ones can be created, viewed, and edited with each of the mobile devices (D1, R1). The shared workspace contains the entire software model and allows the group members to rearrange the CRC cards on screen. Similar to common touch pads integrated in laptops, each mobile device controls a cursor that enables interactions with the shared workspace and the software model (A2).

During role play each group member may own several CRC cards which is visualized by a colored border (cf. Fig. 2). These cards are locked for other users, that is only the owner is able to proceed with the role play when such a class is active (A2). CREWSpace visualizes the progress of the role play by drawing animated delegation edges between collaborating cards. In addition, our tool records all delegation edges that have been created throughout the analysis of a particular use case. Thus, the team may replay a previous analysis and even explore alternatives (D3). When the team is satisfied with the CRC model, CREWSpace allows to draw basic UML relationships between classes to turn the model into a rudimentary class diagram.

4. DESIGN

The design of CREWSpace was inspired by different aspects. First of all, our tool should meet the requirements discussed in Sect. 2. Moreover, also based on these requirements we developed and formatively evaluated CREWW [6, 7], the predecessor of CREWSpace. In contrast to our recent prototype, CREWW leverages Wii-Remotes to allow simultaneous access to the CRC model, but required a single shared bluetooth keyboard for text input.

In order to improve CREWW we asked 26 computer science students to participate in two independent usability studies. In general, our goal was to formatively evaluate CREWW and thus identify additional requirements for a subsequent version. The first study was designed to gather user feedback, in particular, whether the chosen metaphors are accepted, whether interactions with our prototype are intuitive, and whether it is suitable for CRC card based use case analysis. . The second study aimed to gain more insights into how teams actually perform the role play and which features of CREWW would be used regularly..

In general, we found that the participants easily adopted the chosen metaphors and most parts of the controls. However, we can to formulate two additional requirements:

R1: Both studies revealed that users found it tedious to pass around the keyboard or even instruct someone to enter a certain text. *Thus, an improved version requires a way to create and alter CRC cards simultaneously.*

R2: Some participants had problems remembering the functions of less frequently used Wii-Remote buttons, especially when these changed their functions dependent on the current state of the program. *In contrast, software buttons can easily display context-sensitive descriptions such that user can directly recall the current function.*

Next, we briefly introduce the key features not covered in Sect. 3. For detailed information please confer to our previous paper [15].

In order to separate CRC modeling from UML class diagramming, CREWSpace offers two different views. While the Use Case View (Fig. 2) implements the actual CRC method as described in Sect. 2, the UML View allows to add inheritance, association, or aggregation relationships between two classes and, moreover, is able to export this model to edit it with common UML tools (D2).

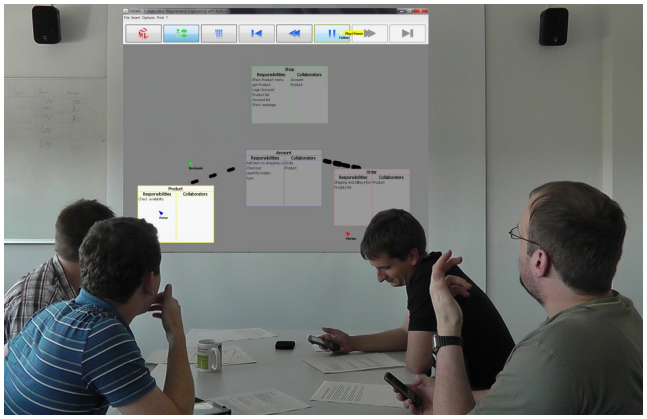


Figure 1: Computer-assisted CRC sessions with CREWSpace

Apart from using mobile devices to interact with the shared software model, they also serve as private workspaces. Hence, every user is able to create or edit CRC cards and use the card overview (Fig. 2) to discuss them with nearby teammates (A3). Also, information about use cases including a use case diagram can be stored in the shared workspace and examined independently from other users via a mobile device (D4).

In a co-located setting it is important that users are aware of what their teammates currently work on. In CREWSpace each group member is always able to recall who is currently editing which class by a small colored, lock-shaped icon on the card (A2). Furthermore, when the state of the role play changes, i.e., the control is given to another group member, it is crucial to notify that person; not only on the shared screen, but also in the private workspace. Besides a short message on the mobile device, it also gives a haptic feedback in order to draw the team member's attention. Furthermore, interactions with the global workspace should be visible to all users. To this end, CREWSpace provides a navigation toolbar with common functions as depicted in Fig. 2.

Finally, digital CRC cards provide an opportunity for automation. For instance, our tool automatically updates all collaborator lists when the name of a card was changed (D1).

5. IMPLEMENTATION

In this section we briefly explain how multi-user interaction in Java is implemented, Android-enabled touch devices are leveraged to control a Java application, and CRC cards can be exchanged between a private and the shared workspace.

Since our prototype is developed for collaborative modeling and role playing and Java does not provide support for multiple cursors natively, we use lightweight UI components (JComponent) as cursors. All cursors are drawn in different colors on the upmost layer of a window and thus will not be covered by other components. In combination with self-defined events they are able to interact with common UI components. Those components that shall interact with any cursor simply implement the according listeners. To restrict simultaneous interaction, we had to modify some of the UI components to establish rules for concurrent access. For example, it is not desired that a user is able to confirm or cancel another user's dialog. Therefore, we extended common UI components such that they can store a user ID. If this ID is already set, i.e., a user interacts with the component, other users are not authorized to access it.

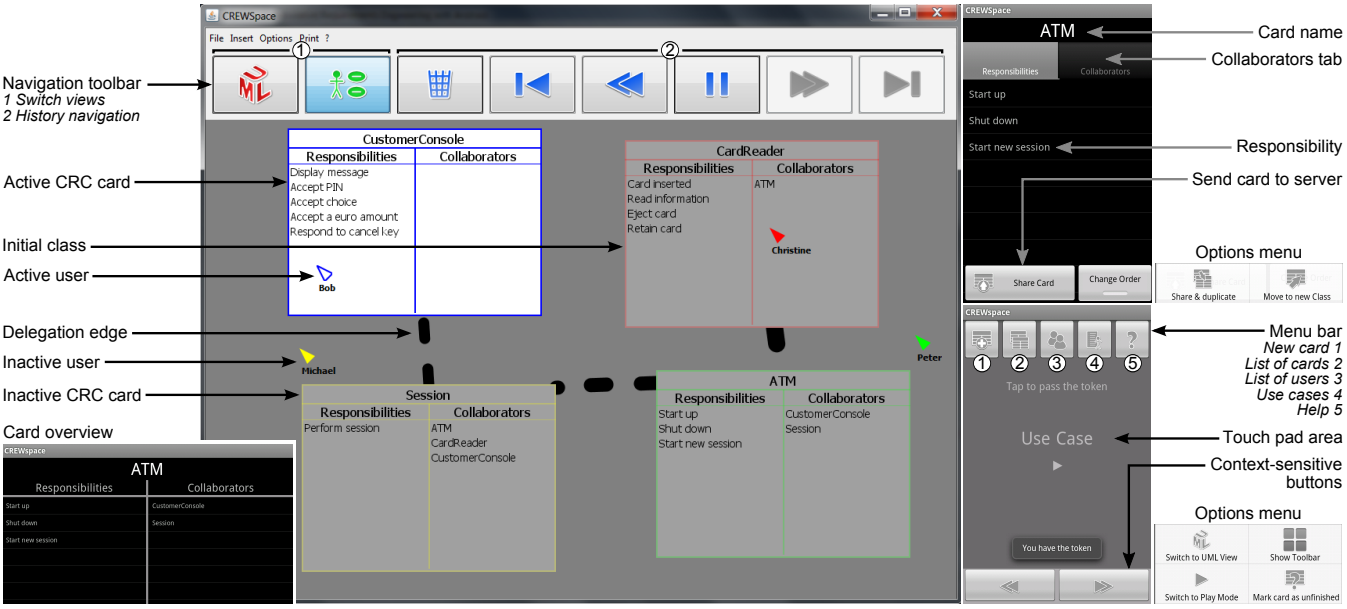


Figure 2: Main application of CREWSpace in Use Case View (center). Edit dialog, main screen (right), and card overview (bottom left) of the mobile application.

The mobile application of CREWSpace provides among others a touch pad feature that allows users to interact with the main application. Using the touch screen of their mobile devices, team members are able to control a colored cursor. Interaction with CRC cards, buttons, and panels is done via different kinds of taps. If such local events are triggered on the mobile device, a UDP connection is used to send messages to the main application where motion or click events are created and forwarded to all UI components that are registered as listeners. In addition to simple taps, the touch pad feature provides a menu bar at the top and two buttons at the bottom of the main screen (Fig. 2). The menu bar enables quick access to commonly used features of the private workspace, for instance, the dialog for creating new CRC cards. The two buttons at the bottom provide context-sensitive features, which trigger additional functions of the main application again using the UDP connection (R2).

Exchanging CRC cards between the private and the shared workspace is based upon a simple client/server approach. Although there are different possibilities to select a class for editing, in general, requests for a CRC card are all treated equally. First, a request message is sent to the main application where all classes are stored in a central data structure. Then the respective card is identified, translated to an XML representation, and sent (via a TCP connection) to the mobile device that requested the class. On the client-side, the XML file is parsed and displayed using the edit dialog (Fig. 2). After modifying the CRC card, it is sent back to the main application and stored in the central data structure. Furthermore, the current view on the screen is updated.

6. EVALUATION

In order to evaluate the usability of our prototype, we conducted a small user study. In general, we investigated if (and how) use case analysis, namely the CRC method, benefits from computer-assisted co-located collaboration.

Our study was designed to be qualitative, i.e, it is not supposed to be complete in terms of statistical significance, it is rather a technique to explore why and how participants behave or acted in a certain situation. This is, we were mainly interested in exploring different ways how CREWSpace is used during collaborative CRC sessions. In particular, we conducted a usability test in combination with a post-study questionnaire. Four teams, in total 15 students, participated in our experiments and were asked to model predefined software systems (cf. Tab. 2). We chose groups with different experience in software design in order to examine how our tool is applied on various skill levels. As CREWSpace shall be integrated in our software engineering course, we mainly recruited undergraduate students. Although all participants were introduced to the CRC method, seven of them had prior experience with CRC cards (cf. Fig. 3), either with our prototype or traditional CRC sessions.

Table 2: Group characteristics

	G1	G2	G3	G4
Students	Graduate	PhD	Undergraduate	
Group size	4	4	4	3
Time spent for modeling	44 min	65 min	40 min	37 min
System to model	online store		light control system	
Initial classes provided	yes	yes	no	no

Each of the team members had access to an Android-enabled touch device with preinstalled mobile application. Also, the groups got descriptions (requirements documents, use case diagrams, etc.) of the systems to model. Earlier studies with our previous prototype revealed that a team might spend a lot of time identifying (and discussing) an initial set of cards. In order to facilitate this process we provided four named but otherwise empty CRC cards for two of the teams. This allowed us to focus on role play and interactions between the team members and with our prototype.

After a short briefing and a warm-up phase to learn the controls and features of our prototype, the teams were asked to model their software system using CREWSpace. To gain more insights into how the teams performed the actual role play and which features were used, we logged all activities including timestamps and user identifiers. Moreover, we videotaped the first two sessions for later analysis. After each experiment we collected feedback from

our participants in short group discussions. Additionally, we asked them to answer a questionnaire, which was based on the assessment principles for usability testing described in [1].

6.1 User Behavior

By analyzing the recorded log-files we were able to generate an overview of the activities of each group over time as depicted in Fig. 3. Here, we classified all activities into five categories as presented in Tab. 3. Please note that activities usually include group discussions and do not always reflect a high interaction with our prototype. To this end, we computed the amount of activities per minute (act/min) logged by our prototype, which is visualized by a black horizontal line for each segment. Role play segments (Category 1), for example, inherently show less activities per minute because role play is mostly based on explanations and group discussions. Editing segments (Cat. 2), on the other hand, describe modifications to CRC cards and thus often reflect a higher amount of activities per minute.

Table 3: Activity categories

Categories	Examples
1 Role play	Calling other classes and fulfilling responsibilities.
2 Editing	Adding, modifying and deleting cards.
3 Navigation	Browse recorded role playing sessions.
4 Change owner	Gaining or releasing ownership of a CRC card.
5 UML Modeling	Adding, modifying, and deleting UML edges.
6 Other	Switching between views, moving CRC cards, etc.

Figure 3 shows that all groups had individual ways of tackling their task. Experiments with our previous prototype strengthen this observation. Some groups, especially the more experienced second group, performed long editing activities (Sections A) along with several discussions before analyzing them in a subsequent role play (B). In contrast, other teams, at a first glance less experienced ones, preferred short editing activities and analyzed these modifications in immediate role playing sessions (C,E). Also, longer editing activities often occurred at the beginning of an experiment because the teams had to create content to work with during role play. This holds especially for groups 3 and 4 as they started to model from scratch. Moreover, groups 1 and 4 resorted to UML modeling in order to discuss and record relationships between classes (D).

Considering the amount of activities per minute (cf. Fig. 3), one can easily spot a difference between groups 1&2 and 3&4. For the first two teams this value is often low (average of 6.4 act/min) as the participants spend a lot of time with discussions about how to improve the software model before they actually modified it. In contrast, groups 3 and 4 (average of 10 act/min) applied more of a *trial-and-error* strategy. Particularly, they changed the software model in advance—sometimes several users even provided individual solutions at the same time (R1). Then, such modifications were discussed and analyzed through role play and afterwards accepted or rejected. The advantage of this strategy is that ideas or solutions are directly visible to all team members. However, it requires some extra effort for editing the CRC cards.

To illustrate the data summarized in Fig. 3, we describe a single example in more detail—general insights are listed below. Please consider the magnified Section E, where Group 1 reset the navigation history in order to start a new role playing session (Cat. 3 segment). Based on the previous role play (Section F) they decided to add a further class to improve the CRC model (Cat. 2) and restructured the layout (Cat. 6). Next, one of the team members took the ownership of the card (Cat. 4) and added obvious responsibilities after a short discussion (Cat. 2). Finally, role playing was used to integrate the new card into the already existing CRC model. During that process the team resorted to the history navigation feature (small Cat. 3 segments) to go back to earlier states of the role play where they replayed important steps (Cat. 1) in order to validate their modifications to the CRC card (Cat. 2) (D3). Finally, Group 1 switched to the UML View (tiny Cat. 6 segment) and used UML modeling to emphasize relationships of the new card with the other classes (Cat. 5) (D2).

Model validation: Replaying a previous step of a use case analysis with the history navigation feature was often used after new classes, responsibilities, or collaborators were added or modified in order to validate such changes to the CRC model. Moreover, the first group also applied the history navigation feature for a final validation of an entire use case scenario (Sections F). To this end, one participant took the lead, went back to the beginning of that scenario, and browsed through the previously finished role play. During that process he briefly explained what all the classes are responsible for or directly asked their owners to do this.

User activity: The chart on the bottom right corner of Fig. 3 depicts the distribution of user activities logged by our prototype. Although there might be a dominating team member, it shows that all people actively participated in a CRC session and the according group discussions. Moreover, team members who used one of our prototypes before seemed to be more confident and even supported unexperienced users.

Editing: Figure 3 does not distinguish between additions and modifications. But our log-files reveal that about 28% of all editing activities were modifications of existing CRC cards. In a traditional setting this could have led to unstructured and messy CRC cards or required unnecessary rewriting effort (D1).

Workspace partitioning: Based on the work of Tse et al. in [25] we investigated if workspace partitioning also occurs in our scenario. At first, when participants had to choose a CRC card for role playing, this decision was often based either on their seating arrangement (mostly left/right) or the current position of their cursor. If there was enough screen space available, newly created cards were often grouped by color. However, during the course of an experiment, the teams changed the layout of the model in order to minimize edge crossings (especially in UML View) or provide additional semantics (e.g., important classes were moved above other ones).

UML modeling: Although we do not feel confident to make a clear statement about the quality of a modeled UML class diagram, groups 1 and 4 used UML modeling in conjunction with role play in order to augment the CRC model with additional information. This may raise the comprehension of the future software system. During earlier experiments with CREWW we observed a similar behavior (D2).

6.2 User Perception

Next, we present the results of our post-study questionnaire. Although the overall user feedback was positive, some participants revealed weaknesses of our prototype and helped us improve its usability. The following paragraphs explain what our participants answered to questions of a particular principle as described in [1].

Suitability for the task: CREWSpace was developed to meet the requirements listed in Sect. 2. Thus, we asked all participants to compare traditional with computer-assisted CRC sessions. Here, most of the drawbacks of the traditional CRC method (D1-D3) were mentioned as advantages of the computer-assisted one. In particular, our participants appreciated that they could easily recall the current state of the role play due to animated delegation edges. The possibility to save their work for later use was men-

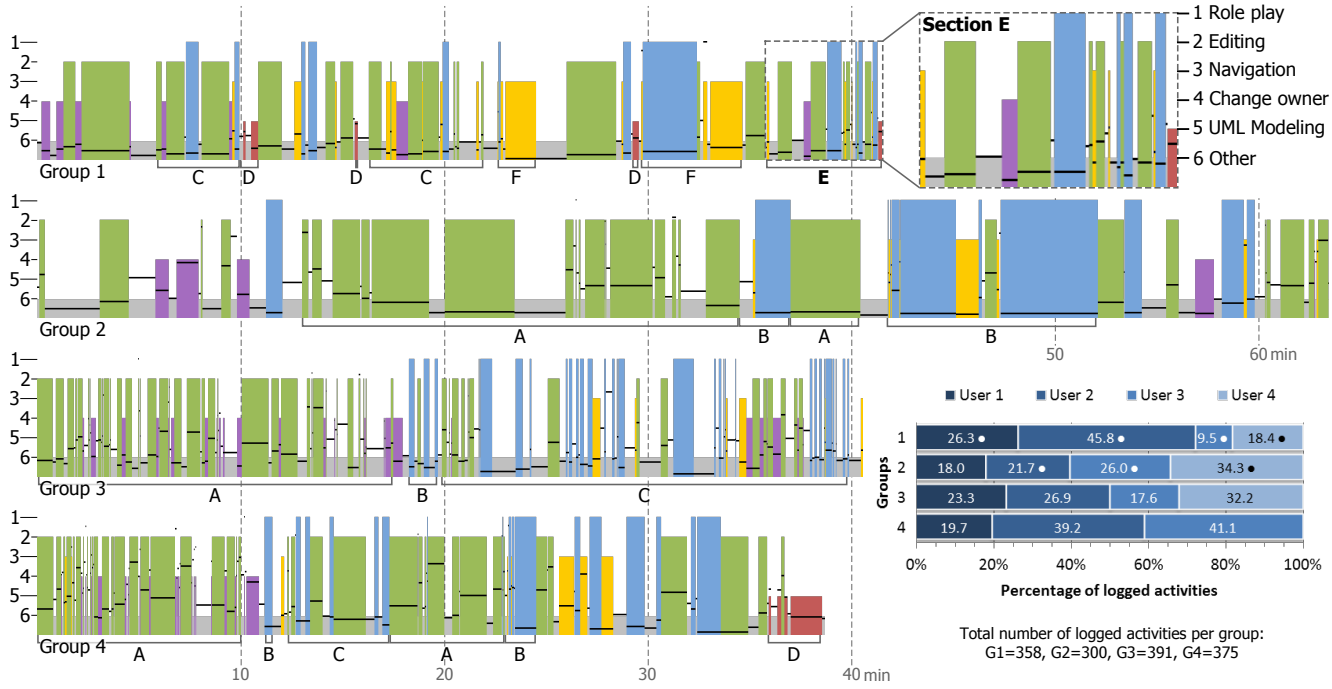


Figure 3: Group activities during the usability study. Represented by color as well as the height of the bars for better readability if printed in grayscale. Horizontal lines depict the amount of activities per minute. The diagram at the bottom left shows the distribution of user activities. A dot • marks users with prior experience with the CRC method.

tioned as an important advantage. Moreover, most participants liked the fact that all team members are able to modify different CRC cards simultaneously with their mobile devices (R1). One person found that computer-assisted CRC sessions could also benefit from further automation features. He suggested to add collaborators automatically to the respective card during role play. Also, the UML modeling features were appreciated (D2).

Our participants also uncovered weaknesses of CREWSpace. For instance, some students found that text input with smaller mobile devices can become tedious and hence, handwritten CRC cards feel more accessible. However, this is a general problem when using small mobile devices and features like word completion try to remedy it. Finally, most students mentioned the extra effort of setting up hardware and familiarize with the controls.

In addition, we asked the students to assess whether computer-assisted CRC sessions yield a better software model. Eleven of them found that this is true especially when people get used to the controls, features, and metaphors of our prototype. Studies with our previous prototype revealed that participants could easily perform computer-assisted CRC sessions after a short time. Three participants thought there will be no significant difference, while only one found that the traditional variant produces a better result because he found it less distracting.

Self-descriptiveness: Since we introduced the students to the controls and features of our prototypes, answers to this part of the questionnaire were mostly short. The participants found that all notifications and tooltips are self-explanatory. Moreover, one student stated that in order to get help he simply asked more experienced users.

Conformity with user expectations: A few students thought that the mobile application might have some issues with respect to different kinds of taps or gestures and the features they represent. Some of these problems were caused by the hardware or the client/server communication, e.g., the touch devices sometimes simply misinterpreted a tap. According to some participants, certain controls or interactions might not be that intuitive or obvious at first. However, there was no overall consensus of how to use certain touch gestures to achieve intuitive controls.

Suitability for learning: Here, the teams mainly suggested a tutorial video that introduces a scenario play through in case no instructor is available. Due to a larger screen of his touch device, one participant found that additional and customizable buttons would improve the comprehensibility of our prototype.

Controllability: Here, we asked if and where participants could imagine more interactivity with our prototype. One student stated that the actual role play seemed to slow down when the team chose to edit a CRC card. In fact, our video recordings reveal that although users are able to edit different CRC cards simultaneously, often only a single or at most two classes need to be adjusted and thus not all participants were active during that time.

Error tolerance: Most of the answers in this section considered detailed suggestions and questions about bugs or errors that occurred during the CRC session and could be fixed afterwards. Nevertheless, most users replied that warning and error messages were self-descriptive.

Suitability for individualization: Here, people asked for a possibility to resize CRC cards individually to obtain more screen space. While this is possible for all cards in advance, we should consider to provide individual or even automatic resizing capabilities. As described before, one participant asked for a way to customize the controls. However, most people do not want to customize their controls at first. Thus, it becomes more important to provide the same intuitive controls for all users.

Support for collaboration: This section of the questionnaire revealed that during role play some participants were not always able to recall directly which user is currently active. Although

this was mostly stated by unexperienced users, we should consider to provide additional feedback techniques through the mobile devices, e.g., simply show the active users name and color on the other devices. Moreover, opinions about collaboration and the restriction of rights were twofold—especially when it comes to editing cards. While some participants suggested that everybody should be able to modify a card, others found that this should only be possible by the owner.

Although we mainly discussed negative feedback that we got from the questionnaire, the usability study suggests that CREWSpace keeps most of the advantages of the traditional CRC method and even compensates its weaknesses. Also, we want to stress that not only this study, but also the experiments with CREWW revealed that it is crucial to support individual strategies and avoid restricting users to a certain workflow.

6.3 Mobile or Gaming Devices?

In our questionnaire we asked the participants who worked with both prototypes (CREWW and CREWSpace) to compare them in terms of controllability. They preferred the mobile devices due to different reasons. Most frequently mentioned was the fact that each team member is able to edit CRC cards. Moreover, our participants appreciated more precise and intuitive controls along with less physical movement. The flexibility of a mobile device was also named as an advantage. For instance, a mobile device is able to display context sensitive and self-exploratory software buttons while a Wii-Remote only provides generic hardware keys.

However, it was stated that mobile devices lose some of the tangibility the Wii-Remotes seemed to preserve. Although in CREWSpace mobile devices may serve as a single card (cf. Fig. 2), our log-files reveal that this feature was rarely used. Hence, we consider integrating the card overview into the role play, i.e., a user that is currently active might simply flip her mobile device and directly view the particular card in the private workspace. A disadvantage of our latest approach might also be that editing cards with a mobile device is hardly visible to all team members (cf. [19, 23]). However, a common strategy was to discuss all required edits and distribute them over the team members if possible. When all modifications where done, the team analyzed the updated model through role play and further group discussions.

6.4 Threats to Validity

During its evolution CREWSpace was repeatedly evaluated in formative user studies (41 participants in total). All these studies including the one presented in this paper were qualitative and task-oriented. They were not meant to quantitatively measure or compare the performance or effectiveness of our tool, but to help improve its usability. We used different methods (video, logging, questionnaire) to compensate for limitations of each individual one and to exploit their respective benefits. The example systems used in our study were quite simple compared to real software. Furthermore, all experiments were limited to a certain time frame and thus none of the teams was able to model the entire system. Both the small number as well as the choice of our participants limit the validity of our results. In longer experiments the groups would have had more time to better adopt the metaphors and the controls of our prototype. Moreover, all of our participants were students and not professional software engineers. People, who deal with system design every day, might have applied CREWSpace differently.

7. RELATED WORK

Support for remote synchronous or asynchronous collaboration is quite common in software development. From simple messenger or video conference systems to software for distributed editing of diagrams such as Gliffy or Creately[1] software engineers and developers can choose from a variety of tools. For instance, D-UML [8] supports distributed software modeling. Users only share the model but no views or windows, which provides a certain flexibility and privacy. Thum et al. [24] follow a lightweight approach for synchronous collaborative modeling. They present SLIM, a UML modeling environment that works in any modern browser and thus lowers the technical entry barriers on the client side. In contrast, our prototype supports co-located collaboration, i.e., users gather in the same room and share the same virtual artifact. Knight [10] and SUMLOW [9] implement co-located collaboration for UML modeling on an electronic whiteboard but do not support CRC sessions or role play.

Although we examined different CRC tools, to the best of our knowledge, there does not exist a tool that supports co-located collaboration for CRC sessions. EasyCRC [20] and CRC Design Assistant [21] were mainly developed to teach and assist students designing object-oriented software. ECoDE [12] is able to generate source code in order to facilitate subsequent development processes. In contrast, Flying Circus [22] uses tree-dimensional CRC cards to further encode different design semantics like implementation complexity or reusability. QuickCRC[2], a commercial tool, supports and automates responsibility driven design of object-oriented software. Besides designing and simulating scenarios, it claims to manage a large amount of cards.

CREWSpace is not the first tool that leverages mobile devices for human-computer interaction. In the late 1990s Myers et al. launched the Pebbles project [18, 17], which comprises different single and multi-user applications controllable by mobile devices. From a technical point of view, PebblesDraw is most closely related to our prototype because, as a multi-user application, it has to deal with issues like managing multiple cursors and concurrent access to UI components. Recently, Yang et al. [27] suggested augmenting a mouse with an interactive touch display in order to provide a customizable secondary workspace. They suggested different applications and showed that unnecessary mouse trips can be prevented. Although multi-user interaction and private workspaces have not been explored in their publication, this approach could bridge the gap between a traditional mouse and a touch device.

We were not the first that used mobile devices for requirements engineering. In contrast to our argumentation that mobile devices are widely available and thus no additional hardware is required (A1), Maiden et al. [16] leverage the mobility of these devices for requirements engineering. Their Mobile Scenario Presenter runs on a PDA and supports requirements engineers while eliciting requirements directly at the workplace.

8. CONCLUSIONS

In this paper we first introduced the traditional CRC method and discussed its advantages and disadvantages. Based upon this discussion we derived requirements for CREWSpace—a tool that assists groups conducting CRC sessions. We briefly presented how such computer-assisted CRC sessions differ from traditional ones, the design of CREWSpace, and important implementation details. However, this paper mainly focused on our user study and gave insights into how our prototype was actually used and how our participants perceived it. The results provide qualitative evidence that CREWSpace supports collaborative computer-assisted CRC sessions and even compensates weaknesses of the traditional variant.

[1] http://www.gliffy.com, http://creately.com (Sept. 2012)

[2] http://www.excelsoftware.com/quickcrcwin.html (Sept. 2012)

Refining the title of this paper, we conclude that for collaborative requirements analysis *virtual CRC cards on smartphones are better than physical CRC cards.*

Although CREWSpace only implements the CRC method, we think that many of the concepts presented in this paper can be transferred to collaborative analyses of other kinds of software models.

9. REFERENCES

[1] EN ISO 9241-110: Ergonomics of human-system interaction – Part 110: Dialogue principles, 2006.

[2] S. Ambler. *The Object Primer: The Application Developer's Guide to Object Orientation*. SIGS Books, 1995.

[3] K. Beck and W. Cunningham. A Laboratory for Teaching Object-Oriented Thinking. In *Proc. of Conference on Object-Oriented Programming Systems, Languages and Applications, New Orleans, LA, USA, October 2-6*, pages 1–6. ACM, 1989.

[4] D. Bellin and S. S. Simone. *The CRC Card Book*. Addison-Wesley Longman, 1997.

[5] J. Börstler. Improving CRC-card role-play with role-play diagrams. In *Companion to the Conference on Object-Oriented Programming, Systems, Languages, and Applications, San Diego, CA, USA, October 16-20*, pages 356–364. ACM, 2005.

[6] F. Bott. Collaborative Requirements Engineering with Wiimotes (CREWW). Master's thesis (in German), University of Trier, Germany, 2009.

[7] F. Bott, S. Diehl, and R. Lutz. CREWW - Collaborative Requirements Engineering with Wii-Remotes (NIER Track). In *Proc. of International Conference on Software Engineering, Waikiki, Honolulu, HI, USA, May 21-28*, pages 852–855. ACM, 2011.

[8] N. Boulila, A. H. Dutoit, and B. Brügge. D-Meeting: an Object-Oriented Framework for Supporting Distributed Modelling of Software. In *Proc. of ICSE Workshop on Global Software Development, Portland, OR, USA, May 9*, 2003.

[9] Q. Chen, J. G. Hosking, and J. C. Grundy. An E-whiteboard Application to Support Early Design-Stage Sketching of UML Diagrams. In *Proc. of Symposium on Human Centric Computing Languages and Environments, Auckland, New Zealand, October 28-31*, pages 219–226. IEEE Computer Society, 2003.

[10] C. H. Damm, K. M. Hansen, and M. Thomsen. Tool Support for Cooperative Object-Oriented Design: Gesture Based Modeling on an Electronic Whiteboard. In *Proc. of Conference on Human factors in computing systems, The Hague, Netherlands, April 1-6*, pages 518–525. ACM, 2000.

[11] E. Gottesdiener. Use Cases: Best Practices. Technical report, IBM, 2003.

[12] K. A. Gray, M. Guzdial, and S. Rugaber. Extending CRC cards into a complete design process. In *Proc. of Conference on Innovation and Technology in Computer Science Education, Thessaloniki, Greece, June 30-July 2*, page 226. ACM, 2003.

[13] I. Jacobson. Object-Oriented Development in an Industrial Environment. In *Proc. of Conference on Object-Oriented Programming Systems, Languages and Applications, Orlando, FL, USA, October 4-8*, pages 183–191. ACM, 1987.

[14] J. W. Kirchner. A Methodological Framework for System Dynamics Model Evaluation. *Dynamica*, 10(1), 1984.

[15] R. Lutz, S. Schäfer, and S. Diehl. Using mobile devices for collaborative requirements engineering. In *Proc. of the International Conference on Automated Software Engineering, Essen, Germany, September 3-7*, pages 298–301. ACM, 2012.

[16] N. Maiden, N. Seyff, P. Grunbacher, O. Otojare, and K. Mitteregger. Making mobile requirements engineering tools usable and useful. In *Proc. of the International Requirements Engineering Conference, Minneapolis-St. Paul, MN, USA, September 11-15*, pages 26–35. IEEE Computer Society, 2006.

[17] B. A. Myers, J. Nichols, J. O. Wobbrock, and R. C. Miller. Taking Handheld Devices to the Next Level. *IEEE Computer*, 37(12):36–43, 2004.

[18] B. A. Myers, H. Stiel, and R. Gargiulo. Collaboration Using Multiple PDAs Connected to a PC. In *Proc. of the Conference on Computer Supported Cooperative Work, Seattle, WA, USA, November 14-18*, pages 285–294, 1998.

[19] D. Pinelle, C. Gutwin, and S. Greenberg. Task analysis for groupware usability evaluation: Modeling shared-workspace tasks with the mechanics of collaboration. *ACM Transactions on Computer-Human Interaction*, 10(4):281–311, 2003.

[20] A. Raman and S. Tyszberowicz. The EasyCRC Tool. In *Proc. of International Conference on Software Engineering Advances, Cap Esterel, French Riviera, France, August 25-31*, pages 52–58. IEEE, 2007.

[21] S. Roach and J. C. Vásquez. A Tool to Support the CRC Design Method. In *Proc. of International Conference on Engineering Education, Gainesville, Florida, October 16-21*, 2004.

[22] A. Savidis, P. Papadakos, and G. Zargianakis. Rapid Visual Design with Semantics Encoding through 3d CRC Cards. In *Proc. of Symposium on Software Visualization, Herrsching am Ammersee, Germany, September 16-17*, pages 193–196. ACM, 2008.

[23] S. D. Scott, M. S. T. Carpendale, and K. M. Inkpen. Territoriality in collaborative tabletop workspaces. In *Proc. of the Conference on Computer Supported Cooperative Work, Chicago, IL, USA, November 6-10*, pages 294–303, 2004.

[24] C. Thum, M. Schwind, and M. Schader. Slim - a lightweight environment for synchronous collaborative modeling. In *Proc. of International Conference on Model Driven Engineering Languages and Systems, Denver, CO, USA, October 4-9*, pages 137–151, 2009.

[25] E. Tse, J. Histon, S. D. Scott, and S. Greenberg. Avoiding interference: how people use spatial separation and partitioning in SDG workspaces. In *Proc. of the Conference on Computer Supported Cooperative Work, Chicago, IL, USA, November 6-10*, pages 252–261, 2004.

[26] C. Wharton, J. Rieman, C. Lewis, and P. Polson. The cognitive walkthrough method: a practitioner's guide. In *Usability inspection methods*, pages 105–140. John Wiley & Sons, Inc., 1994.

[27] X.-D. Yang, E. Mak, D. C. McCallum, P. Irani, X. Cao, and S. Izadi. LensMouse: Augmenting the Mouse with an Interactive Touch Display. In *Proc. of International Conference on Human Factors in Computing Systems, Atlanta, GA, USA, April 10-15*, pages 2431–2440, 2010.