Using Mobile Devices for Collaborative Requirements Engineering

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

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

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

ABSTRACT

In requirements engineering, CRC modeling and use case analysis are established techniques and are often performed as a group work activity. In particular, role play is used to involve different stakeholders into the use case analysis. To support this kind of co-located collaboration we developed CREWSpace, which allows several users to simultaneously interact through Android-enabled mobile devices with the same model displayed on a shared screen. Furthermore, it keeps track of the current state of the role play and, in addition, each mobile device serves as a private workspace; it actually turns into a tangible digital CRC card.

Categories and Subject Descriptors

D.3.1 [**Software Engineering**]: Requirements/Specifications—*Tools*

General Terms

Design

Keywords

Requirements engineering, computer-supported collaborative work, tools and environments

1. INTRODUCTION

In the last decade, novel input devices as well as more powerful mobile devices like smartphones and tablet computers have been successfully used in various domains to augment physical activities (e.g. hiking has been augmented into geocaching) and even to replace low-tech artifacts with virtual ones (e.g. paper calendars have been widely replaced by electronic ones). These virtual artifacts do not only replace their physical counterparts, but most often improve on these by providing additional functionality. Thus, it seems promising to leverage these devices to augment activities and replace artifacts in the software process, as well.

ASE '12, September 03 - 07 2012, Essen, Germany

In software engineering, mental execution or mental walkthrough are typical analysis methods. If several stakeholders are involved, *role play* can be used to perform such an analysis. In these role plays the participants typically take on the role of an element of the model. For example, a participant might play the role of one or more classes. While role play has been applied in software engineering education of most phases, in real software development, role play is mostly used in the requirements phase to bridge the communication gap between developers, customers, and users.

Use cases [5] have become a widespread technique to describe the functional requirements of a system. They capture its externally visible behavior from a user's perspective. Part of their success is due to the simplicity of the graphical notations and their high level of abstraction. Use case diagrams facilitate to involve all stakeholders in the elicitation of requirements and, even to some extent, in their analysis.

While CRC (Class Responsibility Collaboration) cards [2] were initially introduced for teaching an object-oriented way of thinking, the following years showed that CRC cards could also be used for requirements engineering on a professional level. CRC cards model classes, their responsibilities, and collaborators. The CRC method itself may be seen as a transition between the analysis and the design phase of a software engineering process [1]. It includes so called CRC sessions, which involves a group of all stakeholders and facilitate the development of an object-oriented model. After assigning each identified class to one of the group members they play through a use case to uncover inconsistencies and insufficiencies of the current CRC model. In this group work activity it quickly becomes difficult to keep track of which model element is currently active or along which path the current state was reached. Furthermore, some individuals tend to dominate such a discussion, if no moderator enforces the rules of the game.

In order to support a collaborative model analysis as described above, we developed a series of prototypes [3, 11]. Our latest one, CREWSpace [11][1], leverages touch devices (e.g. smartphones or tablets), which run an Android operating system, to enable simultaneous access to a shared virtual artifact. Users can collaboratively create CRC card models, play through use cases, and export them for further use with UML tools. CREWSpace keeps track of the current state of the playthrough, monitors the rules of the game, and thus facilitates fair collaboration between all stakeholders.

In this paper, we introduce CREWSpace, and describe how computer-assisted CRC sessions are executed.

[1] http://www.st.uni-trier.de/crewspace/

Table 1: Comparison of the traditional CRC method and CREWSpace

		Trad. CRC	CREWSpace
R1	No technical support required	✓	×
R2	Supports group work	✓	✓
	Allows for fair collaboration between all stakeholders	✓	✓
R3	Hold CRC cards in the hand, pass them around, or show them to the team members	✓	✓
R4	Easy manipulation of CRC cards, i.e., no discarding and rewriting	×	✓
	Automatically updating collaborators of CRC cards after renaming a card	×	✓
	Duplicate CRC cards without rewriting the entire card by hand	×	✓
	Move responsibilities/collaborators to new class without crossing them out	×	✓
R5	Reuse CRC cards for UML class diagramming	○	✓
R6	Recall the current state of the role play and how it was reached	○	✓
	Maintaining a digital version of the CRC session (CRC cards and scenario playthrough)	×	✓
	Navigate through previous stages of the role play and replay certain parts of the scenario	×	✓
R7	Individual access to a digital copy of the requirements document and the use case diagram	×	✓

2. COMPUTER-ASSISTED CRC

The overall design goal of CREWSpace was to keep the advantages of traditional CRC sessions but enrich them with new features and remedy some of its disadvantages. Mobile devices seemed most suitable to us because they can be leveraged for multi-user interaction with a shared CRC model and also support text input. Moreover, they may serve as a private workspace, i.e., users can view, create, or edit CRC cards independently from their teammates and afterwards share their work with the group.

In order to execute computer-assisted CRC sessions with CREWSpace a computer running the main application and up to ten Android-enabled touch devices with installed client application are needed. We also recommend to either use a large flatscreen or a computer projector to ensure that all team members are able to follow the CRC session. Once all mobile devices are connected to the main application via network the team starts the CRC session.

Phase I: Identifying use cases. Currently, CREWSpace does not completely support the first phase of the CRC method, i.e., team members have to design use cases and scenarios offline. This done, they are able to store any textual information as well as an image of a use case diagram in the main application. During subsequent phases they may access this information through their mobile devices.

Phase II: Identifying classes. Although the actual identification of candidate classes, responsibilities and collaborators is the same, in CREWSpace (digital) CRC cards can be created simultaneously with several mobile devices. By using the Android application team members may enter or change class names and add or change responsibilities or collaborators. Once all digital cards are created, they can be rearranged on screen simultaneously. To this end, each user it is able to control a cursor on the shared screen by using the touch pad area of her mobile device (rf. Figure 2).

Phase III: Analyzing use cases. When a basic set of cards is created, role playing is applied in order to improve the software model. Therefore, all cards have to be distributed over the team members. By claiming a card for herself a team member becomes responsible for it. Such an ownership is visualized by a colored card border. At the beginning of each use case analysis the team chooses an initial class, which is defined to be the active one. The owner of such an active CRC card is in charge to proceed with the scenario playthrough. If the corresponding class is able to fulfill the responsibility on its own, the owner describes that process and completes her turn by clicking at her CRC card. As a consequence, the control returns to the class that called hers and the mobile device of the respective user is vibrating in order to attract attention. On the other hand, if a class is not able to fulfill a responsibility, the owner can delegate a task to another class by clicking on it. As before, the person who is in charge of this card receives a short haptic feedback. This process may be repeated until all responsibilities of the initial class are fulfilled. If required, users can also create new CRC cards and, due to a history function implemented in CREWSpace, navigate through previous stages of their role play and replay certain parts of a scenario.

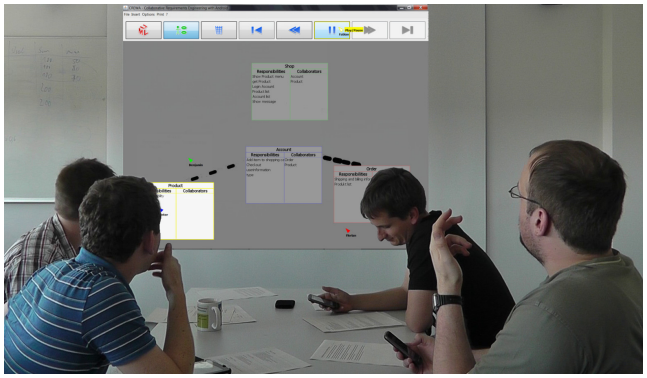

Figure 1: Computer-assisted CRC session with CREWSpace and Android-enabled touch devices.

Transfer to UML. CREWSpace supports modeling of basic relationships between classes in order to transfer the CRC model to a UML class diagram and it is able to export class diagrams for a further use with common UML tools.

3. DESIGN

Based on the requirements in Table 1 we developed different variations of CREWW [3] which leverages Wii-Remotes to allow simultaneous access to the shared CRC model, but required a single shared bluetooth keyboard for text input.

CREWSpace builds upon and improves the features of the main application but leverages mobile devices instead. The following subsections introduce its key features.

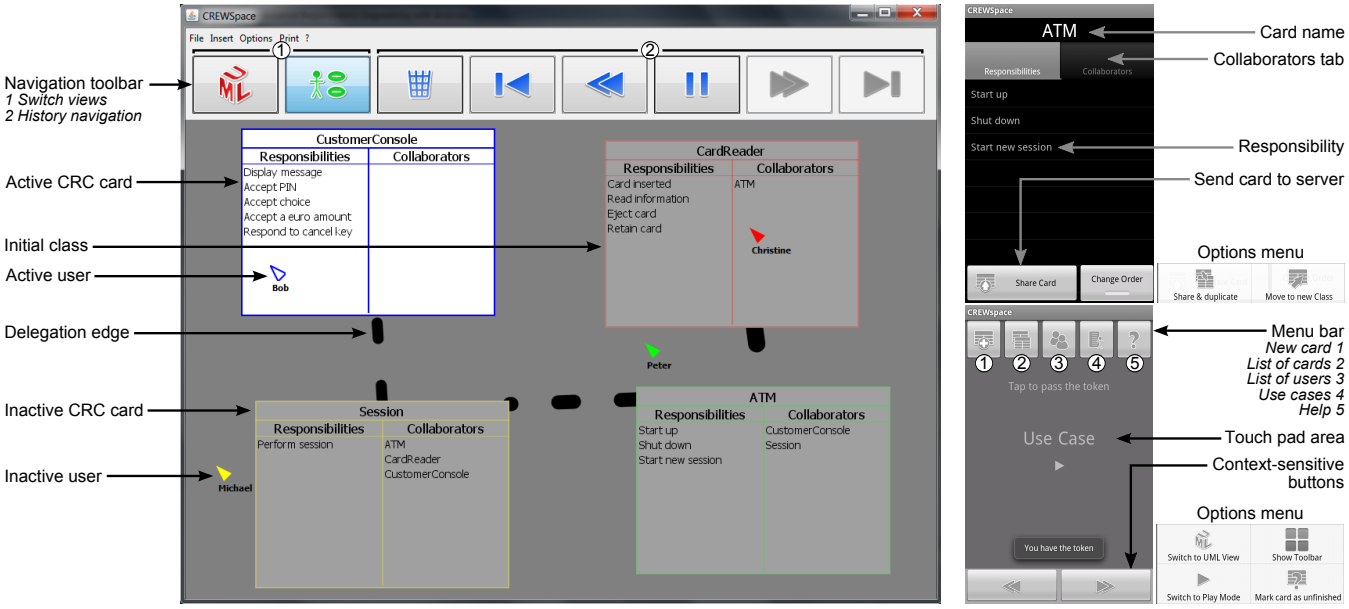

Figure 2: CREWSpace in Use Case View (left). Edit dialog and main screen of the client application (right).

3.1 Views

Our prototype provides two different views: The *Use Case View* (Figure 2), which implements the actual CRC method and visualizes the role play, and the *UML View* to model basic UML class diagrams. Furthermore, in Use Case View users may choose between *Pause* and *Play* Modes for modifying the CRC model and analyzing use cases through role play, respectively. Both views share the same CRC cards, but make use of different relation edges. In Use Case View delegation edges visualize the current state of the role play, while association, aggregation, and inheritance edges are essential for UML modeling.

3.2 Computer-Assisted Role Play

In Use Case View each CRC card may have an owner which is indicated by the color of its border. Only the user that controls the cursor matching this color is able to interact with such a card. Classes that do not have an owner may be occupied by everyone. During the actual role play, only the currently active CRC card is displayed fully opaque indicating that its owner is responsible to proceed with the role play. To visualize delegated responsibilities in Use Case View, we use dotted lines between CRC cards (Figure 2). All visible lines describe the current call stack where their thickness encodes the depth in this stack. In addition, in Play Mode all delegation edges are animated floating from the caller to the callee in order to visualize the current state of a scenario playthrough. Moreover, our prototype records a call history, which contains all delegation edges that were created throughout the whole role playing session. Users may navigate through this history to reinvestigate certain parts of their role play and even stop their investigations at any time and continue from there.

3.3 Private Workspaces

In order to modify a CRC card, in a traditional setting, a team member would typically remove this card from the model, put it in front herself, apply the intended changes, and move it back to its initial position on the table. This example shows that such activities are often performed in a private workspace. In a tabletop setting, Scott et al. investigated different workspaces and found that the zones near each participant, so called personal territories, were mostly used for independent activities [12].

In our setting, each mobile device serves as such a private workspace. Users are able to download CRC cards to it, edit different classes simultaneously, and post all modifications to the shared workspace. In order to prevent multiple users from modifying the same card, which may cause inconsistencies, we implemented a pessimistic locking strategy, i.e., a card can only be accessed by single user. However, team members can still recall who is currently editing a certain class by a colored lock-shaped icon in the upper right corner of a card. The text input itself is independent from our application, i.e. that users are able to choose their preferred input method like on-screen keyboards, simple keypads, or even handwriting recognition. During editing a team member can always flip her mobile device to switch to a widescreen viewing mode where the whole CRC card is displayed like on the shared screen. This mode may be also used to discuss a card with a nearby teammate.

To enable a fast creation of CRC cards, the client application also supports sharing duplicates of a class and moving responsibilities and/or collaborators to a new class. These newly created cards may then serve as starting point to improve or extent the current CRC model. In addition, auto completion allows users to choose collaborators from a list of possible classes rather than typing their full names.

Moreover, in their private workspace team members can access lists of all classes and users along with additional information, read the requirements document and view the use case diagram, or take personal notes. Beyond the features provided by CREWSpace nowadays mobile devices are capable to run different applications like PDF readers or even support collaborative document editing.

3.4 Awareness

Although co-located collaboration inherently comprises a certain level of awareness, in the traditional setting it might be difficult to recall which cards the team members are currently maintaining or editing. To this end, CREWSpace provides a couple of features that can raise the awareness. Some of them were already introduced in Subsection 3.3, but, e.g., when a user switches between Use Case and UML View a prominent message in combination with a short countdown is displayed on the shared screen. Hence, other teammates will directly recognize this activity. People that are currently using their private workspace are able to finish their work and as soon as they return to the main screen of the client application (Figure 2) they can quickly recall the current state of CREWSpace. Moreover, the navigation toolbar provides a possibility to make browsing through the role playing history or switching views transparent to all users. Here, other members can easily follow a teammate when using the buttons of the toolbar.

3.5 Additional Features

Assuming that users do not want to interrupt their current discussion or plan to return to a previous discussion about specific CRC cards later on, they are able to mark classes as unfinished. To this end, CREWSpace allows to add a question mark in the lower right corner of each CRC card. Its color indicates the user who marked the respective card.

With paper-based CRC cards it quickly becomes tedious to keep all cards consistent. Especially, if the name of a class is changed, which collaborates with many other classes. In contrast, CREWSpace automatically updates the according collaborators as soon as the name of a card is modified.

Finally, CREWSpace provides a "Let others have their say" feature. Here, users may vote for teammates that seem to be too dominant during a CRC session. If a certain amount of users share that opinion, the respective team member receives a warning message on her mobile device.

4. RELATED WORK

While there are quite a number of CRC modeling tools, we are not aware of any application that provides support for co-located, collaborative CRC sessions. Nevertheless, we investigated different CRC tools. EasyCRC [8] was originally developed to teach students object-oriented design. It assists students all the way from the beginning of a CRC session to the usage of sequence diagrams for role playing. CRC Design Assistant [9] is another tool that aids students during the design process. Focused on data management it stores CRC models in a database and provides features for effectively creating and maintaining CRC cards. ECoDE [4] supports source code generation. Therefore, classes and scenarios are identified, responsibilities and collaborators assigned, and finally methods and attributes created. In contrast, Flying Circus [10] provides interactive three-dimensional CRC cards for rapid and exploratory software design and, furthermore, augments the CRC model with design semantics like implementation complexity or reusability.

The idea of controlling a computer with mobile or touch-enabled devices has been applied in earlier work. One of the first approaches—the Pebbles project—was launched by Myers et al. in the late 1990s [7]. Over the years they developed and evaluated different single and multi-user applications that augment computers with mobile devices [6].

5. CONCLUSION AND FUTURE WORK

In this paper we introduced CREWSpace—an application that enables collaborative computer-assisted CRC sessions. It leverages Android-enabled devices like smartphones or tablets to allow for multi-user interaction and provide a private workspace. Furthermore, we outlined how CREWSpace can be used to conduct a CRC session, introduced important features of our prototype, and briefly explained how these were realized. CREWSpace was iteratively designed based on user feedback.

While we have only implemented computer-assisted role play for use case analysis, we believe that many of the concepts presented in this paper can be transferred to collaborative analysis of other kinds of software models, for example, to play through sequence diagrams or state charts in combination with object or class diagrams.

6. REFERENCES

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

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

[3] F. Bott, S. Diehl, and R. Lutz. CREWW - Collaborative Requirements Engineering with Wii-Remotes (NIER Track). In *Proc. of International Conference on Software Engineering*, 2011.

[4] K. A. Gray, M. Guzdial, and S. Rugaber. Extending CRC cards into a complete design process. In *Proc. of Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, page 226. ACM, 2003.

[5] I. Jacobson. Object-Oriented Development in an Industrial Environment. In *Proc. of Conference on Object-Oriented Programming Systems, Languages and Applications*, pages 183–191. ACM, 1987.

[6] 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.

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

[8] A. Raman and S. Tyszberowicz. The EasyCRC Tool. In *Proc. of International Conference on Software Engineering Advances*, pages 52–58. IEEE, 2007.

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

[10] A. Savidis, P. Papadakos, and G. Zargianakis. Rapid Visual Design with Semantics Encoding through 3d CRC Cards. In *Proc. of ACM Symposium on Software Visualization*, pages 193–196. ACM, 2008.

[11] S. Schäfer. Android Phones as Input Devices for Groupwork. Bachelor's thesis (in German), University of Trier, Germany, 2011.

[12] S. D. Scott, M. S. T. Carpendale, and K. M. Inkpen. Territoriality in collaborative tabletop workspaces. In *Proc. of ACM Conference on Computer Supported Cooperative Work*, pages 294–303, 2004.