CREWW - Collaborative Requirements Engineering with Wii-Remotes (NIER Track)

Felix Bott
Computer Science Department
University of Trier
Trier, Germany
fbott@laserplussag.de

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

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

ABSTRACT

In this paper, we present CREWW, a tool for co-located, collaborative CRC modeling and use case analysis. In CRC sessions role play is used to involve all stakeholders when determining whether the current software model completely and consistently captures the modeled use case. In this activity it quickly becomes difficult to keep track of which class is currently active or along which path the current state was reached. CREWW was designed to alleviate these and other weaknesses of the traditional approach.

Categories and Subject Descriptors

D.2.1 [**Requirements / Specifications**]; D.2.2 [**Design Tools and Techniques**]; H.5.3 [**Group and Organization Interfaces**]: Collaborative computing

General Terms

human factors, design

Keywords

requirements engineering, collaborative work

1. INTRODUCTION

In software engineering requirements elicitation is an essential process to gather information about the product to develop. Techniques like interviews or questionnaires are often used for this purpose. But as soon as it comes to develop an object-oriented model they are not well suited.

CRC (Class Responsibility Collaboration) cards, introduced by Beck and Cunningham at OOPSLA in 1989 [2], help to involve all stakeholders and facilitate the development of an object-oriented model. Initially designed 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 [13].

Typically, CRC cards are handwritten index cards and model classes, their responsibilities, and collaborators. In a so called CRC session a team of four to seven members with different expertise identifies central classes of the system to build and creates corresponding CRC cards. These cards are then used for role playing, i.e., each team member identifies with one or more classes and acts for them in case their functionality is needed. The goal of this kind of role playing is to evaluate the system in a iterative manner and evolve the classes.

The traditional CRC method is independent from any kind of technical support. Only a set of index cards, pens, and a table is needed. Moreover, it is based on group work, which fosters both productivity and creativity. The execution of a traditional CRC session consists of three subsequent phases [3].

Phase I: The goal of this first phase is to identify use cases and actors in a brainstorming session.

Phase II: Based on the use cases found in the previous phase the team identifies candidate classes along with obvious responsibilities and transcribes them to CRC Cards.

Phase III: During this phase the team investigates all use cases using role play, i.e., each person acts for a certain number of classes. Responsibilities and collaborators are evaluated and new classes might be added.

Transfer to UML: The last step is to translate the CRC model to a more powerful modeling language.

According to Beck et al. one of the biggest advantages of the traditional CRC method is the ability to hold a card in your hand, move it around, or switch it with other cards—simply to identify with a card [2]. Nevertheless, there are also some drawbacks that come along with this approach. Although the initial creation of CRC cards is fast and it is easy to add information, modification or deletion might cause either readability issues or additional time for rewriting a card.

Another problem occurs when using CRC cards for intense role playing. In this phase it quickly becomes non-trivial to keep track of how a certain situation was reached, i.e., one might not be able to recall which classes have been called and in what order.

Moreover, as Ambler [1] put it, "CRC modeling leads directly into class diagramming". But the problem with handwritten cards is obvious. There is no digital version available

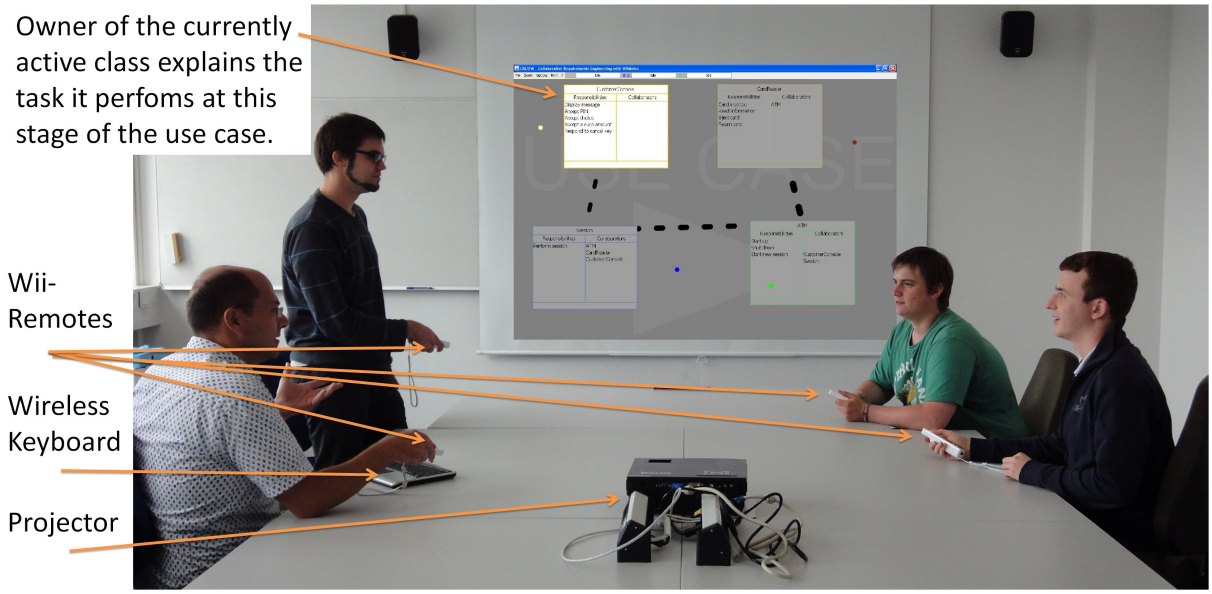

Figure 1: Use-case analysis: The class that is currently active is occupied by the yellow user. Other users are inactive, but can still investigate the process of the scenario playthrough depicted by delegation edges.

neither during nor after the CRC session, which reduces the ease of translating them to UML class diagrams.

The goal of our research is to transfer some of the group work environment into the computer in order to alleviate those drawbacks, but to some extent preserve the tangibility of paper cards. To this end, we use Wii remotes as input devices. For example, when a user wants to move a virtual CRC card, she can literally grab it with the Wii-Remote: by pressing one button with her thumb and one button with her index finger, she virtually holds the card between both fingers and can then move it on the screen. Furthermore, Wii remotes allow for deictic gestures that can be interpreted by the computer as well as other humans. For example, a team member can point at a card on the screen and the other team member can follow the direction of her hand to find the card she is pointing at.

In the rest of this paper, we present the tool CREWW (Collaborative Requirements Engineering with Wii-Remotes) which supports computer-assisted CRC sessions by

- keeping track of the current state and the full history of the playthrough/role play;
- implementing and monitoring the rules and thus to allow for fair collaboration between all stakeholders;
- making the results available in the computer and actually allowing to export them for further use in UML design tools;
- allowing users to add, modify, and delete CRC cards;
- providing an engaging user interface for group work where the current state is immediately visible to all;
- leveraging off-the-shelf gaming input devices to enable simultaneous access to a shared virtual artifact.

2. COMPUTER-ASSISTED CRC

To perform a CRC session with CREWW a computer, up to four Wii-Remotes along with a Bluetooth dongle and the Wii-sensor bar have to be available. We also recommend to either use a large flatscreen or a computer projector to make sure that all team members are able to follow the CRC session. In addition, one may use a Bluetooth keyboard with integrated touch pad and a portable scanner. Figure 1 shows a typical setup.

The Wii-Remote [11] is a single-handed wireless game controller, which besides different digital buttons is equipped with a build-in infrared camera and a three-axis accelerometer. With these components the Wii-Remote is able to determine its position and motion in three-dimensional space and send these data via Bluetooth to a computer.

To receive data from a Wii-Remote in Java we use a client/server architecture that connects with the Bluetooth stack and can be integrated in an arbitrary Java project [6]. The server itself, a C-based program, is a wrapper for a free library called *wiiuse* [12], which is responsible for the connection with several Wii-Remotes and their maintenance. By using this library our server is able to communicate with all Wii-Remotes via the Bluetooth stack and send the gathered information via UDP to a Java client. The design of the Java client is based on the observer pattern [4]. More precisely, the client itself is the subject, which notifies its observers of any state changes by sending `WiiEvents`. Observers are implemented similar to the Java AWT listeners. To be able to react to Wii-Remote actions a class has to implement the respective listener interface, e.g., `WiiButtonListener` or `WiiMotionListener` and process the incoming events. To enable simultaneous interaction with multiple Wii-Remotes we had to modify some of the UI components to establish a basic set of rules for concurrent access.

Using CREWW the phases of a CRC session are performed as follows:

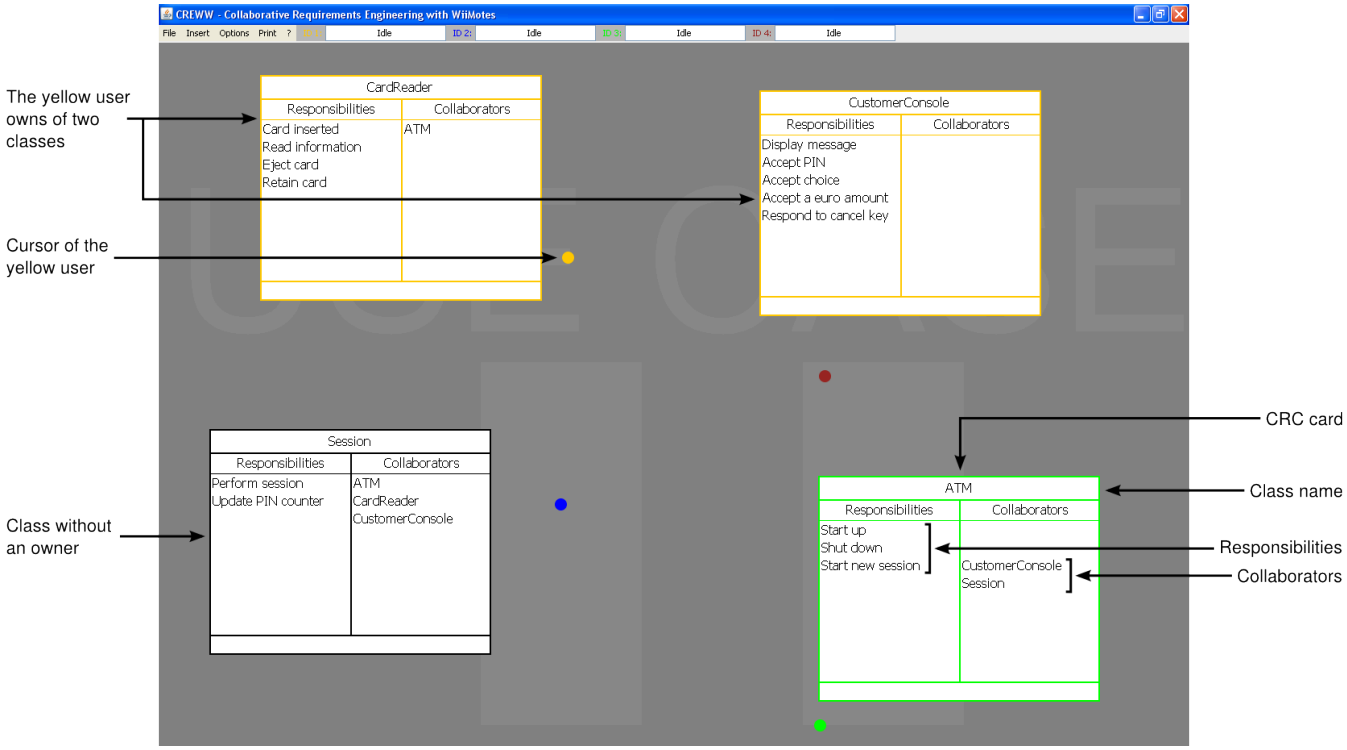

Figure 2: Ownership of CRC cards in CREWW

Phase I: Identifying use cases

Currently, our prototype does not provide support for managing use cases along with different scenarios, or for designing use case diagrams. Thus, the team members have to resort to the traditional method.

Phase II: Identifying classes

This phase differs from the one of the traditional CRC session in the way CRC cards are created. This can be done in two ways:

- The team members can simply create a new (digital) CRC card using the Wii-Remote and the keyboard to fill in class name, responsibilities, and collaborators.
- As an alternative, CREWW provides a printing function for paper CRC cards. These cards are then filled in by hand and read into the computer using the scanning function of CREWW. The scanned images are either shown directly or, if a module for handwriting recognition (ICR) is available, the obtained information can be translated to strings and put into the text fields of a digital CRC card.

Once all digital cards are created, they can be rearranged on the screen. For example, cards can be grouped based on an already defined component structure or based on the person they belong to. Rearranging cards may be done simultaneously by all team members, but each card can only be manipulated by one user at a time.

Phase III: Analyzing use cases

In this last phase the team members start role playing to improve the software model. At the beginning none of the classes has an owner. A team member can now claim one or more classes for herself, which is indicated by a colored border (Figure 2)

Once all classes are distributed, a use case is chosen to be verified. One of the team members activates the initial class for that specific use case.

Only the team member who owns the active class is able to manipulate the control flow and thus proceed with the scenario playthrough. She has two possibilities to move on. Either her class is able to fulfill the responsibility on its own or it has to delegate one or more tasks to other classes. In the former case the owner simply clicks at her own class (by using a pre-defined button of the Wii-Remote) first, to indicate that she has described and thus fulfilled the task and second, to let the control return to the class that called hers. In the latter case the owner activates a class that she needs to collaborate with to solve her task. The team members repeat this procedure until the initial class has fulfilled all of its responsibilities. Figure 1 shows one step of this process. During the role play users may also create additional classes. For each scenario a history of all calls is recorded and might be saved individually. Thus, team members are able to navigate through previous stages of their role play, e.g., if they want to re-investigate a certain part of a scenario to avoid misconceptions. In this manner the team verifies all use cases by playing through different scenarios.

When all use cases have been covered, the actual CRC session is completed.

One of our design goals was to enforce the rules of the role play. It is, for example, important to visually indicate which user is currently allowed to control the scenario playthrough. Furthermore, an owner of a class may delegate a task to a

second one, which then processes this task and afterwards returns the responsibility. It is also crucial for all group members to be aware of the order in which they were involved in the role play or, more precisely, which classes were called. To visualize these kinds of information we had to find a meaningful metaphor.

As shown in Figure 1, during the actual role play only the currently active CRC card is displayed fully opaque indicating that the owner of this card is responsible to proceed with the role play. Furthermore, the inside of her cursor is marked with a white circle.

Animated, dotted edges depict the current state of a scenario playthrough. For instance, if an edge from a CRC card A to a card B exists, the owner of A has delegated responsibility to B. The animation of the edges indicates the direction of the relation: the dots move from the caller to the callee. Additionally, the thickness of an edge encodes the depth of the current call stack, i.e. the more classes were called at a certain point in the playthrough, the thinner an edge is drawn (Figure 1). In case of recursion, animated thinner dots are drawn on top of animated thicker dots. While this may sound awkward, the resulting animation is actually very intuitive.

Users can access the call history through a special menu bar or pre-defined buttons of the Wii-Remotes. Behind the scenes, the class `UseCaseStack` keeps track of both the current call stack and the overall call history. It is important to distinguish between those terms, because the call stack does only contain classes that are currently involved in role play and hides information about former ones. Accessing the call history instead users are able to investigate preceding processes (branches in the dynamic call tree) and if needed automatically replay a certain part of the role play. Furthermore, the users can stop the replay at any time and continue from there.

Transfer to UML

To facilitate the transfer between CRC cards and UML class diagrams CREWW supports the modeling of basic relationships (aggregation, inheritance, association) between classes. Thus, based on the results of the third phase the developers are now able to create a basic UML class diagram, which can be exported to XMI[1]. In addition to saving scenarios, this is another way to make results of CRC sessions reusable.

3. CONCLUSIONS

While there are quite a number of tools for CRC modeling [8, 9, 5, 10, 7], to the best of our knowledge we are not aware of any CRC tool that provides support for co-located, collaborative CRC sessions. CREWW demonstrates that computer-assisted CRC sessions are possible, if the right controls are provided. A quite good decision was to use Wii-Remotes as input devices, because the central idea of the CRC method is role playing and the Wii-Remotes as gaming devices essentially contribute to that. Test persons during the development phase were excited when using our prototype.

Technically, the integration of Wii-Remotes including the synchronization of GUI components was a quite interesting challenge that resulted in a client/server architecture, which manages the concurrent access and can be reused for other computer-assisted kinds of model analysis, for example, to play through sequence diagrams in combination with object diagrams.

[1]XML Metadata Interchange—standard format for exchanging metadata information between UML tools

4. 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 *Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA'89), New Orleans, Louisiana*, pages 1–6. ACM, 1989.

[3] D. Bellin and S. S. Simone. *The CRC Card Book.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1997.

[4] E. Gamma, R. Helm, R. Johnson, J. M. Vlissides, C. Larman, and N. M. Wilkinson. *Design Patterns - Elements of Reusable Object-oriented Software.* Addison-Wesley Longman, Amsterdam, NL, 1995.

[5] K. A. Gray, M. Guzdial, and S. Rugaber. Extending CRC cards into a complete design process. In *8th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education, ITiCSE 2003, Thessaloniki, Greece*, page 226. ACM, 2003.

[6] S. Müller. A Java-API for Multi-User Interaction with Wii-Remotes. Master's thesis (in German), University of Trier, Germany, 2008.

[7] QuickCRC - Responsibility Driven Design. `http://www.excelsoftware.com/quickcrcwin.html`, Mar 2010.

[8] A. Raman and S. Tyszberowicz. The EasyCRC Tool. In *Second International Conference on Software Engineering Advances (ICSEA 2007), August 25-31, 2007, Cap Esterel, French Riviera, France*, pages 52–58. IEEE Computer Society, 2007.

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

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

[11] Nintendo Wii. `http://wii.nintendo.com`, 2010.

[12] Wiiuse - The Wiimote C Library. `http://www.wiiuse.net/`, Dec 2009.

[13] N. M. Wilkinson. *Using CRC Cards: An Informal Approach to Object-Oriented Development.* Cambridge University Press, Cambridge, UK, 1999.