Using Role-Play Diagrams to Improve Scenario Role-Play

Software Engineering for Software-intensive Systems

Claudia Schumacher
E-Mail: clasch@mail.upb.de
Motivation

- After gathering requirements a software design needs to be created
- Very critical step for the success of the software system
- There are different approaches to develop a software design
- CRC-cards are used as informal tool for supporting object-oriented development
- This approach has two main problems: bad documentation and mixing up classes and objects
- In order to solve these problems role-play diagrams are created
Table of Contents

1. CRC-Cards
2. Role-Play Diagrams
3. Example using CRC-cards and role-play diagrams
4. Summary
5. Commentary
What are CRC-cards?

CRC-cards

- A CRC-card is a index card which corresponds to a class
- CRC stands for **Class**, **Responsibilities** and **Collaborators**
- introduced by Kent Beck and Ward Cunningham 1989
- created as an approach to teach object-oriented design

<table>
<thead>
<tr>
<th>Class:</th>
<th>Responsibilities:</th>
<th>Collaborators:</th>
</tr>
</thead>
</table>

A CRC-card

Ward Cunningham

Kent Beck
CRC-Cards

CRC-Card Session (1/2)

Preparing for a session

- Group size of 4-6 people with experts of the domain, object-oriented design and programmer
- Select a scenario with focus on a small and manageable part of the system

The creation of the card is the important aspect of the CRC-card technique not the cards themselves
Perform a session

- Creating cards:
  - during a brainstorming each member of the group suggests needed classes for the scenario
  - obvious responsibilities and collaborations can be added

- Playing the scenario:
  - each participant is responsible for at least one card
  - the participant takes their classes as role
  - by walking through the scenario the group is simulating what the system will do by playing their roles and interact with each other
  - during the play missing responsibilities and collaborations are added to the class
Problems

- The CRC-cards represent classes but they are used as objects during the scenario.
- Communication between classes are defined but the chronological order is not documented.
- After a scenario play the whole process is lost because of missing documentation.

Role-Play Diagrams (RPD)
Role-Play Diagrams (RPD)

- Are used to document object interaction during the scenario play
- The objects in the RPD are instances of the classes modelled by CRC-cards
- RPD covers the most important aspects from UML object and collaboration diagrams – but they are simpler and less formal

*design goal: simplicity*
Objects in RPD are represented by object cards
Object cards represent an instance of a CRC-card
An object card is showing the name, class and properties relevant for the current scenario

<table>
<thead>
<tr>
<th>Class: Borrower</th>
</tr>
</thead>
<tbody>
<tr>
<td>Responsibilities:</td>
</tr>
<tr>
<td>Knows name</td>
</tr>
<tr>
<td>Knows borrowed books</td>
</tr>
<tr>
<td>Knows total fines</td>
</tr>
</tbody>
</table>

| Collaborators: |

Borrower0 : Borrower
name = paul
borrowed books = 2
total fines = 0
Structure

- Objects which collaborate are connected by a line
- Communication is only allowed during connected object cards
- Being collaborators only means that there can be communication
Structure

- Interaction between objects are documented on the connecting lines between the communicating objects
  - Numbering scheme keep track of the order
  - Small description
  - Arrow denotes the direction
  - Annotation documents information that is returned

```
<table>
<thead>
<tr>
<th>Borrower0 : Borrower</th>
<th>book0 : Book</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>title = Who are you?</td>
</tr>
<tr>
<td></td>
<td>on loan = yes</td>
</tr>
<tr>
<td></td>
<td>1: on loan?</td>
</tr>
<tr>
<td>→</td>
<td>yes</td>
</tr>
</tbody>
</table>
```

Structure

- As soon as the knowledge of an object changes, the corresponding object card is updated

```
book0 : Book

<table>
<thead>
<tr>
<th>title</th>
<th>Who are you?</th>
</tr>
</thead>
<tbody>
<tr>
<td>on loan</td>
<td>yes, no</td>
</tr>
</tbody>
</table>
```
**CRC-Cards and Role-Play Diagrams**

An example of an object-oriented analysis with CRC-cards and role-play diagrams

- identify candidate classes
- filter the list of candidates
- create CRC-cards for the remaining candidates
- allocate responsibilities to CRC-cards/classes
- define scenarios to test/evaluate the model
- prepare the scenario role-play
- perform the actual role-play
- record scenarios
- update CRC-cards and scenarios
Introduction to the Example

- Technical library system for a university department
  - Searching and lending of technical library materials, including books, videos, and technical journals
  - All items
    - can be borrowed for a specific time according to the item type
    - have according to their type a different fine
  - Each borrower can borrow up to 10 items if
    - he has no overdue lendable items
    - he has total fines less than 100
Step 1: Identify Candidate Classes

- Generate a list of candidate classes that might be of interest for the problem
- Can be done by brainstorming
- Can be done by noun extraction

- **Technical library system for a university department**
  - Searching and lending of technical library materials, including books, videos, and technical journals

---

Technical library, a university department, searching of, lending of, books, videos, technical journals
Step 2: Filter Candidates

- Reduce the number of candidates to a manageable size

**Technical library, a university department, searching of, lending of, books, videos, technical journals**

**Technical library:** seems to refer to the stock or collection of things
⇒ keep library

**University department:** outside the problem domain
⇒ Discard

**searching of** and **lending of:** Verbs used as nouns
⇒ Responsibilities
⇒ Discard

Diagram:
- identify candidate classes
- filter the list of candidates
- create CRC-cards for the remaining candidates
- allocate responsibilities to CRC-cards/classes
- define scenarios to test/evaluate the model
- prepare the scenario role-play
- perform the actual role-play
- record scenarios
- update CRC-cards and scenarios
Step 3: Create CRC-Cards
Step 4: Allocate Responsibilities

Class: Library
Responsibilities:
- Knows all available lendables
- Search for lendable
Collaborators:
- Lendable

Class: Lendable
Responsibilities:
- Knows its due date
- Knows its title
- Knows if late
- Check out
Collaborators:
- Date

Class: Date
Responsibilities:
- Knows current date
- Can compare to dates
- Can compute advance dates
Collaborators:

- identify candidate classes
- filter the list of candidates
- create CRC-cards for the remaining candidates
- allocate responsibilities to CRC-cards/classes
- define scenarios to test/evaluate the model
- prepare the scenario role-play
- perform the actual role-play
- record scenarios
- update CRC-cards and scenarios
Step 5: Define Scenarios

- Define the scenario in detail
- It is important that the scenario is defined well
- Scenarios are test cases for the CRC-card model
- Start with simple scenarios

**John Doe will borrow the book 1984.**

- He is a registered borrower
- He has no outstanding fines
- The book is available and not on loan
Step 6: Prepare Scenario Role-Play

- Everyone agrees the CRC-card description
- One team member act as scribe during the play
- Each member is responsible for a selection of CRC-cards
- Each member will act out for all objects of his cards
- Initial RPD is drawn
Example Initial RPD

userAgent

book2 : Book
  title = Farm der Tiere
  on loan = no

book1 : Book
  title = 1984
  on loan = no

theLibrarian

theLibrary

borrower1 : Borrower
  name = John Doe
  borrowed books = none
  total fines = 0

borrower2 : Borrower
  name = Jane Smith
  borrowed books = none
  total fines = 0
Step 7: The Actual Role-Play

- Simulate how the future system would work according to the CRC-card model
- All acting is restricted to the responsibilities on the cards
- Only the card holder of the active object is allowed to act!
Step 8: Record Scenarios

**UserAgent:**

*I request from theLibrarian that the book with title '1984' is checked out to borrower „John Doe“.*

Control is transferred to theLibrarian.

```
userAgent

1: check out „1984“ to „John Doe“

theLibrarian
```

- identify candidate classes
- filter the list of candidates
- create CRC-cards for the remaining candidates
- allocate responsibilities to CRC-cards/classes
- define scenarios to test/evaluate the model
- prepare the scenario role-play
- perform the actual role-play
- record scenarios
- update CRC-cards and scenarios
Step 9: Update CRC-Cards and Scenarios

- Missing classes, responsibilities or collaborators can be added
- Classes, responsibilities or collaborators can be removed or changed
- This also happens during the play
- If a lot of classes are changed the role-play should start again
CRC-Cards Define Structure

- CRC-Cards define structure

<table>
<thead>
<tr>
<th>Class: Borrower</th>
</tr>
</thead>
<tbody>
<tr>
<td>Responsibilities:</td>
</tr>
<tr>
<td>Knows name</td>
</tr>
<tr>
<td>Knows borrowed books</td>
</tr>
<tr>
<td>Knows total fines</td>
</tr>
<tr>
<td>Collaborators:</td>
</tr>
<tr>
<td>Book</td>
</tr>
</tbody>
</table>

```
Borrower
+name : string
+totalFines : double

Book
+title : string
+on loan : bool
```
Role-Play Diagrams Define Behavior

- RPD provide a clear definition of behavior between classes
Summary

- CRC-Cards are a common approach to collaborative object-oriented analysis and design
- CRC-Cards force the participants to reason about models and explain design decisions
- Role-Play Diagrams are a helpful extension for the CRC-card approach
- Role-Play Diagrams help to document the CRC-card scenario play
- Role-Play Diagrams help to point out the difference between classes and objects
- It is easy to transform the informal CRC-cards and role-play diagrams into a formal design document
I would recommend to use role-play diagrams additional to CRC-cards to teach object-oriented thinking

- They clarify the difference between classes and objects
- But all participants should work with role-play diagrams
Commentary

If a CRC-card approach is used to create an OO-Design I would recommend the additional use of role-play diagrams

- RPD document the analysis process
- RPD can be easily transformed to a formal diagram
- PRD enable the possibility to reproduce the processed scenarios even for non-participants
- Extended numbering should be used for attributes
Thank you for your attention!

These are some of the first CRC-cards ever made by Ward Cunningham.
Bibliography

JÜRGEN BÖRSTLER: Using Role-Play Diagrams to Improve Scenario Role-Play, Graph Transformations and Model-Driven Engineering, Springer, 2010


JÜRGEN BÖRSTLER: CRC-cards and Roleplay Diagrams Informal Tools to Teach OO Thinking