



# A Pattern for an Effective Class Responsibility Collaborator (CRC) Card

---

Format, Process and Interaction

By:

Mohamed E. Fayad, Haitham Hamza, and Huáscar Sánchez



# CRC Card Origin - Overview

---

- In 1989, Kent Beck and Ward Cunningham introduced the notion of CRC cards at the annual OOPSLA conference.
- The CRC-Cards technique original purpose was to be used in either practical software development or in Object Oriented (OO) education.
  - In development, CRC-Cards can offer valuable insights during the early stages of development.
  - Another benefit of CRC Cards is in teaching Object Concepts in programming language courses.
- Several reported case studies have demonstrated the effectiveness of CRC Card as a tool for introducing OO programming concepts and for improving the understanding of objects/classes.

# Our Pattern Solution Objective

---

- To discuss the main problems with current CRC-Cards.
- To propose a new look at CRC-Cards that try to avoid most of the problems presented in traditional cards.
- To identify the potential shortcomings of current CRC-Cards technique and to propose a pattern that can be followed to write an effective CRC-Cards.





# Current CRC Cards - Problems

---

- Current CRC-Cards lack some essential qualities that might affect the effectiveness of the developed system that uses them. The main problems in current CRC-Cards can be summarized in the following points:
  - *Possibility of Low Cohesion and High Coupling*
  - *Macho Classes*
  - *Exclusion of Services*
  - *No Clear Role is Defined*
  - *Difficulty in Defining Responsibilities*
  - *Difficulty in Mapping*
- These problems are overcome in our proposed CRC Card format. Allowing the achievement of great flexibility for analysts, designers, and developers to ease the sharing knowledge and decision making processes.



# Writing an Effective CRC Card is Hard

---

- For CRC-Cards to enhance the development of systems, main essential quality factors should be satisfied. A summary of these main points or quality factor are mentioned herein:
  - A major advantage of the CRC-Cards tool resides in its simplicity to understand. However, In Current CRC Cards such simplicity may not convey all the required information needed in the following steps in the development:
    - No information about the type of Collaboration there is.
    - In other words, the card does not specify the services that its class offers to the other classes that collaborate with it. And
    - Having too much information in a CRC-Card might preclude them from being widely applied.
  - It is important to understand the role of each class in the system in order to identify its responsibilities. However, current CRC-Cards do not provide a way to differentiate between the different roles of the same class.
  - Defining the class responsibility is crucial for developing an accurate class diagram and latter an effective system.
    - Identifying all these responsibilities in one CRC-Card might create great confusion for the developers.
    - it becomes confusing to match a responsibility of a class to another collaborating class in the system.



# Current CRC Card

---

The current CRC Card is presented herein.

Class	
Responsibility	Collaboration



# Comparative Study

---

To understand the differences between our proposed solution and existing work. We provide a comparative study of both structures.

Our Pattern Solution	Existing Work
<ol style="list-style-type: none"><li>1. Explicit Role</li><li>2. Unique Responsibility.</li><li>3. Explicit exhibition of Services.</li><li>4. Role is closely related to Responsibility.</li><li>5. Few Responsibility-focused Services.</li><li>6. Simple Fill-out workflow.</li></ol>	<ol style="list-style-type: none"><li>1. None</li><li>2. Multiple Responsibilities per Class.</li><li>3. Responsibilities and Services are considered to be the same. No explicit exhibition.</li><li>4. No Relationship. Role is missing.</li><li>5. There is not such distinction.</li><li>6. Complex, almost confusing, fill-out workflow.</li></ol>

# A Closer Look at our solution



- We present an enhanced representation for CRC cards as a solution to some of the problems in current CRC-Cards.
  - The new version will include a clear role for each class which will aid in the discovery of superclasses and their respective subclasses.
  - This class role will also be useful when defining the class responsibility.
  - Each class will be allowed to have only one unique responsibility.

## Multiple Responsibilities?

What should we do?



## Solution:

If more than one responsibility is identified, additional classes should be formed.

**Result:** When limiting responsibilities to one per class we are helping to prevent low cohesion and high coupling as well as reduce the possibility of macho classes.



# A Closer Look at our solution



- The revised CRC card will include the services offered by each class.

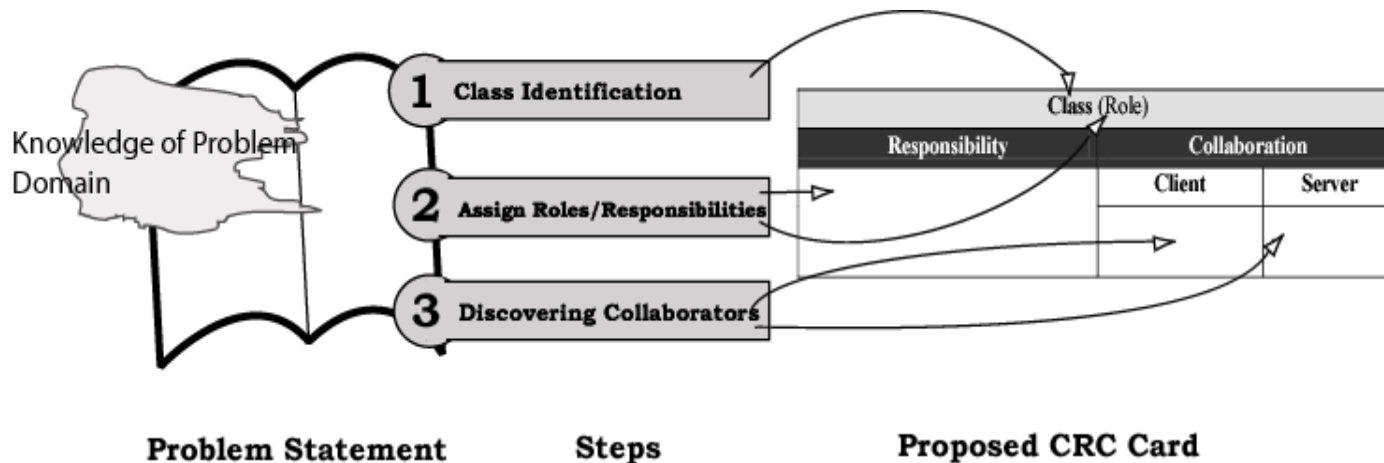
## **Benefits:**

1. To help verify the validity of the class responsibility.
  2. To ensure that overlapping functionality is avoided.
- Our proposed CRC Card Format:

Class (Role)		
Responsibility	Collaboration	
	Client	Server

# Creating the Proposed CRC Card

- Important Steps:
  1. Class Identification.
  2. Assigning Roles and Responsibilities.
  3. Discovering Collaborators.
- These steps are illustrated in a more general way as follow:





# Applicability of the Proposed Format

---

- In this section we show how the proposed CRC-Card format can be applied in a simple example.

**Problem statement:** The services that are linked with the word *Genealogy* have been growing tremendously across the Internet landscape. This idea has been touched by several online businesses, such as Ancestry.com, Msn, etc., but it is still in its infancy. We propose a simple Family Tree design. This system will offer a central storage device, where all the information of current members will be stored. Each member will have full control of his/her information portrayed in the system.

The system will provide consistent updating, searching and tracking mechanisms to facilitate an easy interaction between the system and the members throughout the entire Family Tree. An efficient user-friendly navigation mechanism will be presented as well. This will guarantee full access to all the features of the system. Members of the system will be able to share this experience by inviting new users to either start up their own family tree, and/or enroll in a member's family tree. The latter would happen in the case that these potential members are relatives of an already enrolled member. Regarding guests of the system, there will be a section exclusively for them, allowing them to visit current Family trees with certain limitations. They can also create their own family tree if they desire.



# Applicability of the Proposed Format

---

- Applying the main steps of our proposed pattern:
  1. *Step 1: Identifying Classes:* The Class identification process does not vary in both approaches. Both approaches use similar techniques along with the overall knowledge on the subject from analysts, and designers. Similar naming conventions are applied, except they vary when dealing with compound nouns. Compound nouns are treated as one word using “camel-casing”.

<b>FamilyTree (Role)</b>		
<b>Responsibility</b>	<b>Collaboration</b>	
	<b>Client</b>	<b>Server</b>



# Applicability of the Proposed Format

---

- Applying the main steps of our proposed pattern:
  2. *Step 2: Assigning Roles and Responsibilities:* The proposed CRC Card, different from the current format, focus on defining and exhibiting the class role within the system. The presence of a well-defined role makes things easier for the analyst because each role is tightly bound to a unique responsibility. For Responsibilities “No naming rules” are required in this step. However, for understanding purpose, analysts need to define clear and cohesive responsibilities.

<b>FamilyTree (FamilyTree)</b>		
<b>Responsibility</b>	<b>Collaboration</b>	
Illustrate the bond of a group of people	<b>Client</b>	<b>Server</b>



# Applicability of the Proposed Format

---

- Applying the main steps of our proposed pattern:
  3. *Step 3: Discovering Collaborators:* For this, we use grammatical parse over the problem statement, looking for verbs and/or verbs phrases. These verbs or verb phrases usually corresponds to the methods used to fulfill a unique responsibility of a class. For method declarations we use “camel-casing” *except* that the first letter is in lowercase. This methods are nested within the “Server” compartment of our proposed CRC Card. Along with explicitly inclusion of these services, several classes that communicate with one particular class will be placed on the left compartment of the Collaboration section. This section is called *Clients*.

<b>FamilyTree (FamilyTree)</b>		
<b>Responsibility</b>	<b>Collaboration</b>	
To show relationships between people	<b>Client</b>	<b>Server</b>
	Member Family	initiateTree() connectFamily() joinToTree() search()



# Conclusion

---

- We have provided a new enhanced format of the CRC Cards.
- We have offered a solution to overcome the built-in problems of current CRC Cards.
- The high-level process of writing the proposed card format is also presented, through a simple application example.



The End

---

Thank You!